UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

PROCESADO PARALELO DE DATOS DE ACELERÓMETROS PARA DETERMINAR NIVELES DE ACTIVIDAD FÍSICA

Luis Arroyo Lozano
Tutor: Iván González Martínez

FEBRERO 2015

“Dedicado a mi tío Cayetano”

RESUMEN

El proyecto de investigación Up&Down está realizando un estudio de la actividad física, comportamiento sedentario y salud en niños y adolescentes. Para ello, entre otras actividades, están tomando datos de acelerometría de más de 1000 niños y adolescentes.

Dichos datos se toman mediante acelerómetros que los niños y adolescentes llevan puestos durante una semana, y que generan medidas cada 10 segundos. A partir del análisis de los datos, se debe poder discriminar el nivel de actividad física desarrollado por cada individuo y su patrón de ruptura del sedentarismo.

Se utilizan dos tipos de acelerómetros, los primeros calculan el movimiento únicamente en el eje vertical, mientras que los segundos lo calculan en los tres ejes así como la intensidad de la luz y otros parámetros que no usaremos.

Como los datos ya estaban tomados, analizaremos solo el eje vertical, ya que es el eje común a ambos acelerómetros. Con esto evitamos generar distintos resultados dependiendo de un acelerómetro u otro.

La cantidad de información a procesar es muy grande (más de 100.000 medidas por individuo, y una población de más de 1000 individuos), por lo que se plantea la necesidad de realizar un procesado de los datos utilizando técnicas de programación paralela. En este proyecto, se ha usado OpenMP como técnica para paralelizar el procesado de datos.

El objetivo consiste en implementar de manera paralela el algoritmo que procesa los datos de los sujetos, determinando su actividad física y el nivel de sedentarismo así como mostrar una serie de gráficas para el posterior análisis estadístico.

PALABRAS CLAVE

Actividad física

Acelerometría

Acelerómetro

Análisis

Cuartiles

Paralelismo

SBBI

Sedentarismo

ABSTRACT

The research Project Up&Down is conducting a study of physical activity, sedentary behavior and health in kids and teens. For this purpose, they are taking accelerometry data from over a thousand kids and teens.

These data are taken with accelerometers that kids and teens had been wearing during a week. These accelerometers generate measures every ten seconds. We have to discriminate the physical activity and the sedentary behavior of each person from the data analysis.

We use two accelerometer types. The first one calculates movement just on vertical axis while the second one calculates movement on the three axes as well as the light intensity and other parameters we will not use.

As the data were already taken, we will analyze just the vertical axis since is the common axis for both devices. With this, we avoid generating different results depending on a device or another.

The amount of information to be processed is too large (over a hundred thousand measures for each person and a population over a thousand people), so the need for data processing using parallel programming techniques arises. In this project, OpenMP has been used as a technique to parallelize the data processing.

The purpose is to implement a parallel algorithm that processes people data and determines their physical activity, sedentary level and shows a series of graphs for further statistical analysis.

KEYWORDS

Accelerometry

Accelerometer

Analysis

Parallelism

Physical activity

Quartiles

SBBI

Sedentary Activity

ÍNDICE

Tabla de contenido

ÍNDICE DE FIGURAS

Tabla de ilustraciones

1. INTRODUCCIÓN

El estudio de la actividad física y de los hábitos de los niños y adolescentes es un tema bastante actual debido a la cantidad de aplicaciones y dispositivos que la tecnología proporciona. La moda por monitorizar la actividad física está en auge, ya sea para adultos, adolescentes o niños.

1.1. MOTIVACIÓN

Los estudiantes de la facultad de Deportes han seleccionado al azar a más de 1000 sujetos para analizar su actividad física y sus niveles de sedentarismo para obtener información sobre las rutinas que siguen los niños y adolescentes hoy en día y para ello, se han puesto en contacto con nosotros.

1.2. OBJETIVOS

La finalidad de este proyecto, aparte de poder monitorizar la actividad física de los sujetos, es poder categorizar a los mismos según su índice de ruptura del comportamiento sedentario. Para esto, primero se analizarán los sujetos y se filtrarán siguiendo una serie de requisitos en sus datos recogidos. Una vez filtrados, se generará una gráfica para cada sujeto que mostrará su índice de ruptura (SBBI) así como unas graficas globales indicando la categorización de los sujetos y el porcentaje de sujetos válidos y no válidos.

1.3. PLANIFICACIÓN

En una primera reunión con los compañeros de la facultad de Deportes, se hablaron sobre los requisitos que debería tener la aplicación. Poco tiempo más tarde, se terminaron de ajustar en un documento de análisis de requisitos. Una vez obtenidos, se comenzó la fase de diseño de la aplicación y su posterior codificación.

Una vez realizada una primera versión de la aplicación, se concertó una reunión para mostrar los resultados y proponer algún cambio o ajuste en el programa. Tras esa segunda reunión, se especificaron más algunos requisitos y se procedió a su revisión. Al terminar estos ajustes, se contactó con el tutor para que mostrase el resultado final a los estudiantes de Deportes.

1.4. ESTRUCTURA DEL DOCUMENTO

En los siguientes apartados se mostrarán los requisitos pedidos por los clientes y se explicará que tipo de tecnología y metodología se ha utilizado así como las decisiones tomadas en cuanto al diseño, implementación y estructuración del programa final. Por último se mostrarán las pruebas realizadas, resultados obtenidos y las conclusiones de los mismos.

2. ESTADO DEL ARTE

A lo largo de este apartado se explicarán las propuestas y algoritmos necesarios para el filtrado de los sujetos así como de la determinación de los niveles de intensidad para este proyecto. Estas propuestas han sido proporcionadas por los compañeros de la facultad de deportes implicados en este proyecto.

2.1. FILTRADO

El filtrado de sujetos es la parte más importante de este proyecto, ya que un filtrado incorrecto puede generar estadísticas alteradas ya que los valores recogidos no serán verídicos. En este apartado se explicarán detalladamente los tipos de filtrado utilizados y su finalidad para este proyecto.

TIEMPO NO VESTIDO

Para este filtro, se utiliza el algoritmo de Choi[1]. Este algoritmo se basa en varios criterios para indicar si el sujeto sigue llevando el acelerómetro puesto o se lo ha quitado. Se considera entonces que el aparato no está puesto cuando se cumpla alguna de estas condiciones:

- Existen intervalos en los que las medias generadas por el acelerómetro son siempre cero. Estos intervalos deben ser de al menos noventa minutos. A este criterio se le denomina ventana de ceros consecutivos y viene a decir que el aparato se ha quitado y se ha dejado en una mesa o similar.

- Existen intervalos de poco tiempo (de dos a cinco minutos) en los que el acelerómetro genera valores mayores que cero. Estos intervalos se deben encontrar entre dos intervalos de ceros consecutivos de máximo treinta minutos cada uno. Este intervalo corto se denomina tiempo máximo de caída mientras que los intervalos de ceros consecutivos antes y después del tiempo de caída se llaman ventanas entre movimientos artificiales.

- Con este criterio se evita tener en cuenta cualquier movimiento que se pueda producir de manera artificial mientras el dispositivo esta quitado, como por ejemplo dejarlo en una mesa y al cabo de un rato, cambiarlo de sitio.

DIAS VÁLIDOS

Se dice que un día es válido siempre y cuando se cumplan una serie de horas completas en las que el sujeto ha llevado puesto el acelerómetro. Estas horas no pertenecen a un estándar, por lo que en cada estudio se decidirán las horas mínimas necesarias para la validez del día.

La validez de estas horas es determinada por el algoritmo explicado en el apartado anterior.

SUJETO VÁLIDO – NO VÁLIDO

Al igual que con los días válidos, no existe un estándar en la elección del número mínimo de días para concluir que un sujeto es válido o no. Es cierto que para poder conseguir una estadística acorde con la realidad y eliminar cualquier sujeto sospechoso de su validez, se deberá ser muy estricto con este parámetro, pero también se deberá evitar ser muy restrictivo, ya que en ese caso, la población valida será muy reducida frente a la inicial.

También es importante diferenciar entre días de diario o fines de semana, ya que los sujetos pueden no tener el mismo estilo de vida u horarios.

2.2. NIVELES DE INTENSIDAD

Al igual que pasa con el filtrado, se deberá establecer una serie de puntos de corte para diferenciar las intensidades de la actividad física (explicados en el análisis de requisitos). Estos puntos de corte no son estándares, sino que se han seleccionado acorde con el tipo de estudio y los objetivos que se desean alcanzar.

Dentro de las intensidades, habrá que diferenciar entre intensidad sedentaria, actividad física ligera, moderada y vigorosa. Adicionalmente, se plantea eliminar las partes de actividad física moderada y vigorosa con el fin de identificar el índice de ruptura del sedentarismo.

3. TECNOLOGÍA UTILIZADA

El desarrollo del proyecto está hecho en su integridad en un entorno Linux de 32 bits, concretamente en Ubuntu en su versión 14.04. El hardware usado es una maquina con un procesador dual-core y 2GB de memoria RAM.

Como lenguaje de programación se ha usado C por su versatilidad y comodidad ya que durante toda la carrera se ha ido aprendiendo este lenguaje así como técnicas para depurar el estilo y la optimización del código.

Los ficheros generados por los acelerómetros tienen la extensión "csv". Este formato tiene similitud con el Excel de Windows. Contendrán los diferentes valores obtenidos en columnas.

Adicionalmente se han usado en este proyecto ficheros temporales y scripts Bash para obtener las imágenes de las gráficas de los sujetos.

Como método para paralelizar el código se ha utilizado OpenMP, ya que tiene ciertas ventajas sobre MPI o CUDA. El compilador GCC ofrece soporte para OpenMP desde la versión 4.7, por lo que al compilar simplemente hay que poner en el makefile la cláusula "–fopenmp" para ejecutar con este método.

Otra ventaja es que no necesita paso de mensajes como en MPI ni hardware especifico como ocurre con CUDA. OpenMP ha sido un método utilizado en la asignatura de arquitectura de sistemas paralelos y en el que se hizo bastante hincapié, por lo que la soltura es mayor que en los otros métodos citados.

En este caso, se ha optado por una paralelización de grano fino. El modelo de programación seleccionado así como el rendimiento obtenido será explicado mas adelante.

Para las pruebas realizadas durante el desarrollo del proyecto se ha usado la misma máquina y un número reducido de sujetos para simplificar la depuración y reducir los tiempos de ejecución.

Una vez terminado el desarrollo, se ha procedido a realizar pruebas de rendimiento en entornos con diferentes versiones del software y en máquinas con más capacidad y numero de

procesadores para poder obtener datos estadísticos de la aplicación. Estas estadísticas se mostrarán en el punto 5 de este documento.

4. ANÁLISIS DE REQUISITOS

A continuación se expondrán los requisitos funcionales y no funcionales obtenidos tras varias reuniones con los alumnos y profesores que componen el proyecto de Up&Down.

4.1. REQUISITOS FUNCIONALES

FILTRADO DE "TIEMPO NO VESTIDO"

Se tiene que aplicar siempre. Se utiliza el algoritmo de Choi (2011) que contempla tres criterios de filtrado:

a) Ventana de tiempo "ceros consecutivos"

b) Tiempo máximo de caída

c) Ventana de entre movimientos artificiales

Esos tres valores deben ser parametrizables. Los parámetros de tiempo no vestido deben ser eliminados para análisis posteriores.

FILTRADO DE "DÍAS VÁLIDOS"

Un día válido es aquel que cumple un determinado tiempo de "tiempo vestido" desde que empieza el día hasta que acaba. Ese valor debe ser parametrizable

DETERMINACIÓN DE SUJETO "VÁLIDO-NO VÁLIDO"

Un sujeto es válido para el estudio cuando tiene un determinado número de días válidos. Existen dos opciones:

a) Sin distinguir días de diario y fin de semana

b) Distinguiendo días de diario y fin de semana

La opción deber ser elegible y los parámetros parametrizables.

PUNTOS DE CORTE DE DETERMINACIÓN DE NIVELES DE INTENSIDAD

Analizar los puntos de corte de intensidad de AF según las categorías:

- Sedentarismo
- Actividad Física Ligera
- Actividad Física Moderada

- Actividad Física Vigorosa

Estos valores deben ser parametrizables.

FILTRADO DE HORAS

Incorporar la posibilidad de limitar el análisis solo para determinados días de la semana (L, M, X, J, V, S, D) y una franja horario determinada. Este filtrado deber ser elegible (realizar/no realizar) y parametrizable en día semana (L a D) y hora (de 00:00 -23:59 h). Esta opción no sería imprescindible pero sí muy deseable.

FILTRADO DE TIEMPO DE AF MODERA-VIGOROSA

Debe existir una opción en la que se pueda eliminar del análisis de la señal aquellas partes de la misma que sean consideradas AF MODERADA-VIGOROSA.

Debe poder parametrizarse un "BOUT" mínimo para la ruptura del tiempo sedentario con intensidades correspondientes a AF de INTENSIDAD-MODERADA a vigorosa sea considerada y eliminada del análisis de la señal (el valor por defecto será "cero minutos")

ANÁLISIS DEL PATRÓN DE INTERRUPCIÓN DEL TIEMPO SEDENTARIO

Se trata de crear un indicador continuo que mediante un valor indique el nivel de "agrupamiento-interrupción" del patrón de comportamiento sedentario de forma que a través de ese indicador se puedan distinguir patrones de comportamiento. Podemos denominar a ese indicador "Sedentary Behavior Break Index" (SBBI).

CATEGORIZACIÓN DE SUJETOS

Los sujetos deberían poder ser caracterizados por percentiles de puntuación. Para este proyecto, se usarán Terciles.

4.2. REQUISITOS NO FUNCIONALES

ARCHIVOS A UTILIZAR COMO ARCHIVOS DE ENTRADA

Para el desarrollo del este trabajo se utilizarán como archivos de entrada archivos ".csv" en "modo 0" (solo aceleraciones verticales) con un epoch de 10 segundos. La duración del estudio para este trabajo será de 7 días.

TRABAJO POR LOTES Y ORGANIZACIÓN DE LOS RESULTADOS

El software debería de ser capaz de trabajar los archivos por lotes y dar resultados sobre un conjunto de sujetos.

Los sujetos analizados deberían poder obtenerse en un archivo de forma que se puedan manejar en Excel y trasladar fácilmente a SPSS para su tratamiento estadístico.

ARCHIVOS GRÁFICOS DE SALIDA

Ver la posibilidad de incorporar alguna gráfica sobre el patrón de interrupción del tiempo sedentario o sobre el conjunto de sujetos.

MANUAL DE USUARIO

Sería necesario elaborar un pequeño "set de instrucciones" que permita usar la aplicación a un investigador que no la conozca.

PARALELISMO

Todo el procesamiento y análisis de los sujetos debe ser ejecutado de manera paralela para optimizar y mejorar el rendimiento del programa ya que son muchos los sujetos que se pretenden analizar.

5. DISEÑO Y DESARROLLO

En este apartado se explicará cómo se ha desarrollado la aplicación: su estructura, TADs utilizados y el método de paralelismo elegido. Así mismo, se mostrará cómo ha sido diseñada la interfaz de usuario, el fichero de configuración y las salidas generadas por el programa.

5.1. ARQUITECTURA

A continuación se mostrará el esquema con las entradas, salidas y TADs utilizadas para la implementación de la aplicación así como una explicación de cada estructura utilizada.

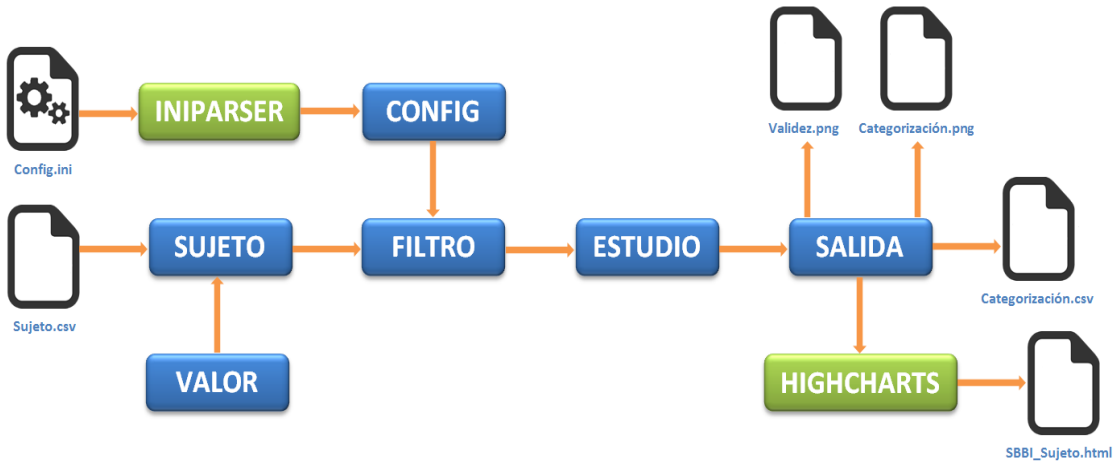


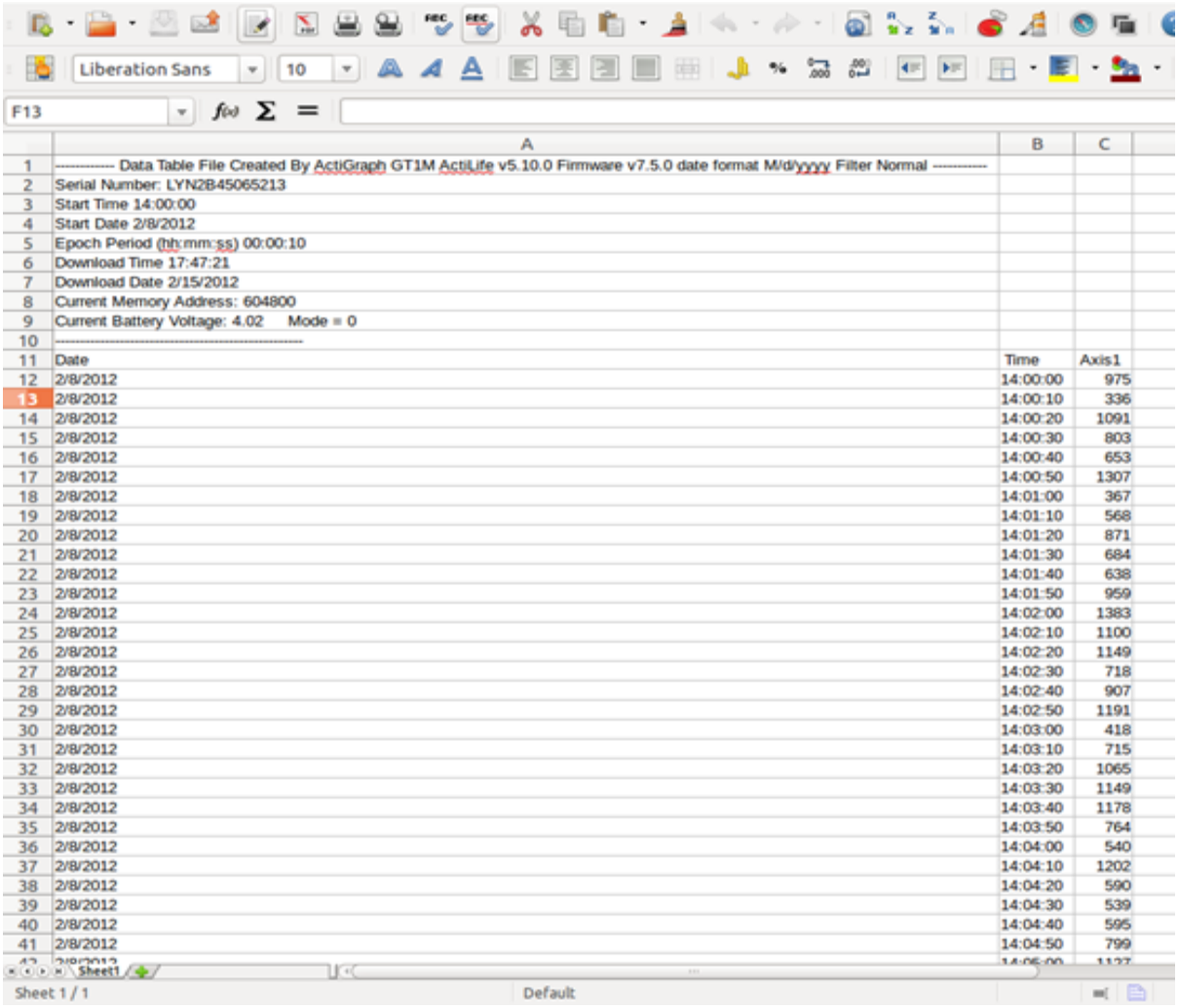
Ilustración 1. Esquema de funcionamiento

FICHEROS DE CONFIGURACIÓN Y SUJETOS

Como se puede ver en la imagen, se utiliza un fichero de configuración con los valores de las variables que el usuario define para el filtrado de sujetos y los niveles de intensidad. Este fichero debe encontrarse en el mismo directorio que el ejecutable. En caso de no encontrarse, se mostrará un mensaje de error. Más adelante se explicará el formato elegido para el fichero de configuración.

Además del fichero de configuración, deben estar presentes los ficheros de los sujetos a analizar. Cada sujeto se lee desde un fichero que contiene en una tabla la fecha, hora y valor del acelerómetro en ese instante así como una cabecera que contendrá información sobre el acelerómetro, la fecha y hora de inicio del estudio y la configuración de los parámetros. El

número de filas de la tabla puede variar dependiendo de esta última configuración. A continuación se muestra un ejemplo del fichero de un sujeto.



	A	B	C
1	---------- Data Table File Created By ActiGraph GT1M ActiLife v5.10.0 Firmware v7.5.0 date format M/d/yyyy Filter Normal ----------		
2	Serial Number: LYN2B45065213		
3	Start Time 14:00:00		
4	Start Date 2/8/2012		
5	Epoch Period (hh:mm:ss) 00:00:10		
6	Download Time 17:47:21		
7	Download Date 2/15/2012		
8	Current Memory Address: 604800		
9	Current Battery Voltage: 4.02 Mode = 0		
10	----------		
11	Date	Time	Axis1
12	2/8/2012	14:00:00	975
13	2/8/2012	14:00:10	336
14	2/8/2012	14:00:20	1091
15	2/8/2012	14:00:30	803
16	2/8/2012	14:00:40	653
17	2/8/2012	14:00:50	1307
18	2/8/2012	14:01:00	367
19	2/8/2012	14:01:10	568
20	2/8/2012	14:01:20	871
21	2/8/2012	14:01:30	684
22	2/8/2012	14:01:40	638
23	2/8/2012	14:01:50	959
24	2/8/2012	14:02:00	1383
25	2/8/2012	14:02:10	1100
26	2/8/2012	14:02:20	1149
27	2/8/2012	14:02:30	718
28	2/8/2012	14:02:40	907
29	2/8/2012	14:02:50	1191
30	2/8/2012	14:03:00	418
31	2/8/2012	14:03:10	715
32	2/8/2012	14:03:20	1065
33	2/8/2012	14:03:30	1149
34	2/8/2012	14:03:40	1178
35	2/8/2012	14:03:50	764
36	2/8/2012	14:04:00	540
37	2/8/2012	14:04:10	1202
38	2/8/2012	14:04:20	590
39	2/8/2012	14:04:30	539
40	2/8/2012	14:04:40	595
41	2/8/2012	14:04:50	799

Ilustración 2. Fichero de Sujeto

Para que el programa funcione acorde con la configuración de los acelerómetros, se deberán ajustar los parámetros del fichero de configuración para que éstos coincidan. El directorio de los ficheros se puede insertar directamente desde el programa principal.

TAD MANEJOFICHEROS

Todos estos ficheros de entrada se procesan desde el módulo de manejoFicheros. Este módulo se encarga del procesado de los ficheros y los directorios necesarios mediante las siguientes acciones

- Crea una estructura de tipo CONFIGURACION en la que se almacenarán los parámetros leídos del fichero de configuración.

- Lee y parsea el fichero de configuración mediante el uso de una librería externa. Esta librería se basa en crear un diccionario que crea y almacena ciertos parámetros del mismo tipo agrupados en el fichero con etiquetas. Esta estructura sirve para una mejor comprensión por parte del usuario.
- Restaura el fichero de configuración en caso de pérdida o mala gestión de los valores de los parámetros.
- Crea una lista con los nombres de los ficheros se encuentren en el directorio proporcionado. Estos ficheros deben tener el formato adecuado para ser reconocidos como ficheros válidos, en otro caso, se desecharán y no se añadirán a la lista.
- Dada una fecha, este TAD calcula el día de la semana en el que cayó (Lunes, Martes...)[2].

TAD SUJETO

Cada fila válida del fichero contiene la fecha, hora y valor del acelerómetro, por lo que este TAD implementa una nueva estructura de tipo VALOR que almacenará cada tripleta de estas variables.

Esta lista de valores se almacena en otra estructura de tipo SUJETO junto con el nombre del fichero y el tamaño de la lista.

Para poder cargar los sujetos en estas estructuras, se usa una función que recibe el directorio en el que se encuentra el fichero y el nombre del fichero a leer. Antes de comenzar con la inserción de tripletas en la estructura, se salta la cabecera del fichero. Al finalizar la lectura, se devuelve el sujeto cargado.

Dentro de este TAD existen otras dos funciones encargadas de la creación y la liberación del sujeto. La primera reserva memoria para un sujeto e inicializa sus variables mientras que la segunda libera la memoria utilizada.

TAD FILTRADO

Este módulo se encarga de determinar si un sujeto es válido o no mediante el análisis de los valores leídos. Este análisis se lleva a cabo mediante la implementación de los algoritmos de filtrado (tanto el tiempo no vestido como los días válidos) explicados en el apartado del estado del arte y las variables definidas por el usuario en fichero de configuración que se ha leído previamente.

La función de filtrado se ha separado en otras funciones para facilitar el proceso y hacer más fácil la lectura del código. Se ha creado una función que devuelve si un día es válido

o no y otra que permite obtener el número de horas no vestidas. Esta última función necesita dos funciones auxiliares para analizar primero el tiempo de ceros consecutivos y posteriormente el tiempo de movimiento artificial. Estas dos funciones generan una nueva señal con los intervalos no vestidos eliminados de la señal original. Con esto, se mejora el rendimiento en funciones posteriores ya que reduce considerablemente el tamaño de la señal de cada sujeto.

El filtrado genera una nueva lista de valores para cada usuario, eliminando los tiempos y días no válidos. Esta nueva lista será con la que se trabajará a partir de ahora.

Los sujetos analizados, ya sean válidos o no, se almacenan en nuevas estructuras de tipo DIA y ESTUDIOS que se usarán posteriormente.

TAD ESTUDIO

Este TAD consta de una estructura de tipo DIA que contiene la fecha, valores de actividad sedentaria, ligera, moderada y vigorosa (en minutos), el SBBI, la lista de valores de día y el tamaño de la misma. Estos días se generan por cada día válido y se añaden a la estructura ESTUDIO.

Esta estructura contiene una lista de días válidos, el número de días almacenados y las medias de actividad física sedentaria, ligera, moderada y vigorosa así como la media del SBBI de esos días. También tiene dos campos extra para la categorización y validez del sujeto.

Este TAD se encarga de calcular la actividad física de un sujeto válido mediante el análisis de sus CPM. Estos CPM se obtienen mediante una función que genera un valor medio por cada minuto de señal. Una vez obtenido el valor CPM, se analiza en esa misma función para obtener el tipo de actividad física realizada en ese intervalo de tiempo, comparándolo con los valores establecidos en el fichero de configuración.

Este proceso se realiza para cada día válido almacenado en la estructura ESTUDIO. Al finalizar el análisis de todos los días, se calculan las medias de actividad física y se almacenan en las variables creadas para ello.

Al igual que pasa con los sujetos, existen otras funciones para la creación y eliminación de estudios. La primera, reserva memoria para las estructuras e inicializa los valores de actividad física y las medias para evitar usar variables con valores aleatorios dentro. En cambio, la segunda, libera la memoria reservada anteriormente para evitar la saturación de memoria.

TAD SALIDA

Este módulo se encarga de generar los ficheros de salida con los resultados obtenidos durante el análisis de sujetos.

Como se aprecia en el esquema mostrado anteriormente (Ilustración 1), existen diversas salidas: una por cada sujeto válido, una tabla con los sujetos analizados y su SBBI diario, ordenados por la media total de éste, una gráfica indicando el ratio de sujetos válidos y no válidos y otra con los sujetos agrupados por la categorización. El contenido de dichos ficheros se mostrará en el apartado siguiente.

Para la creación de la tabla, se ha implementado una función que imprime la cabecera nada más comenzar la ejecución y otra que inserta las filas de los sujetos ya ordenadas. Una vez terminada la escritura de sujetos, se cierra el fichero.

Par cada sujeto válido se imprime una gráfica que muestra el SBBI durante el tiempo que dura el estudio. Esta gráfica se hace mediante la escritura de un fichero de tipo "HTML" con una serie de etiquetas que llaman a otra librería externa[3] capaz de generar la gráfica. Al abrir el fichero, se abre una página web que permite al usuario ver la gráfica y descargarla en varios formatos. Los datos de cada sujeto se ordenan según el día de la semana para que exista una normalización entre los formatos de los ficheros.

Por último, se crean las gráficas de la categorización y el índice de validez. Para la primera se ha implementado una función que calcula la categorización de los sujetos a partir de la media total del SBBI. Esta categorización genera dos puntos de corte, agrupando los sujetos en tres grupos del mismo tamaño (terciles). Para obtener los puntos de corte, se hace una sencilla operación matemática con el SBBI mínimo y máximo, mientras que para obtener los índices de validez y no validez simplemente se cuentan los sujetos.

Una vez separados por terciles, se crean dos ficheros temporales con instrucciones que usará el programa GNU-Plot[4]. Estos nuevos ficheros temporales crearán en el directorio principal las gráficas de la categorización y validez de sujetos y luego serán eliminados.

5.2. DISEÑO

Dentro de este apartado, diferenciaremos entre el diseño de los ficheros de entrada y los de salida. No se analizarán los ficheros de los sujetos, ya que su diseño es generado automáticamente por los acelerómetros y ya se ha comentado en el apartado anterior.

ENTRADA: FICHERO DE CONFIGURACIÓN

Este fichero se ha diseñado siguiendo la librería externa para la lectura de configuraciones. Esta librería permitía el uso de etiquetas o secciones para poder diferenciar y agrupar las variables y hacer más fácil la lectura y configuración por parte del usuario.

Dentro de cada sección, se escribe el nombre de la variable seguido del valor que se quiera indicar. En el caso de introducir cadenas, deberá indicarse el tanto el inicio como el final mediante comillas dobles ("..."). Para finalizar una línea, se utiliza el símbolo de punto y coma (;).

Si el usuario desea escribir un comentario sobre la variable o alguna indicación, puede hacerlo después de este punto y coma, ya que la librería lo tomará como comentario y no será procesado.

El final de sección se reconoce mediante la lectura de otra sección o cuando se llega al final del fichero. Un ejemplo de fichero de configuración sería el siguiente:

config.ini

```
1
2 [filtradoSujetos]
3 duracionEstudio = 7;
4 cerosConsecutivos = 60;
5 tiempoCaida = 2;
6 movimientoArtificial = 30;
7 horasVestidas = 10;
8 diasValidos = 3;
9 diasValidosDiario = 3;
10 diasValidosFinde = 1;
11 opcion = 1;
12
13 [determinacionActividadFisica]
14 countsSedentario = 99;
15 countsLigero = 1999;
16 countsVigoroso = 4000;
17
18 [filtradoActividadFisica]
19 opcionFiltrarAF = 0;
20 diaFiltrado = 6;
21 horaInicio = 17:00;
22 horaFinal = 18:00;
```

Ilustración 3. Fichero de configuración

SALIDA: RATIO DE VALIDEZ DE USUARIOS

Esta grafica sirve para mostrar de manera visual la participación de los usuarios en el estudio mediante su validez. Se cuentan los sujetos válidos y los no válidos mientras se van analizando y una vez terminado, se genera esta gráfica mediante la herramienta GNU-PLOT[5].

Este tipo de graficas es bastante útil, ya que permite ver rápidamente si los requisitos para poder determinar la validez de un sujeto son demasiado restrictivos o permisivos y poder ajustarlos en base a las necesidades. Este es un ejemplo de fichero de validez:

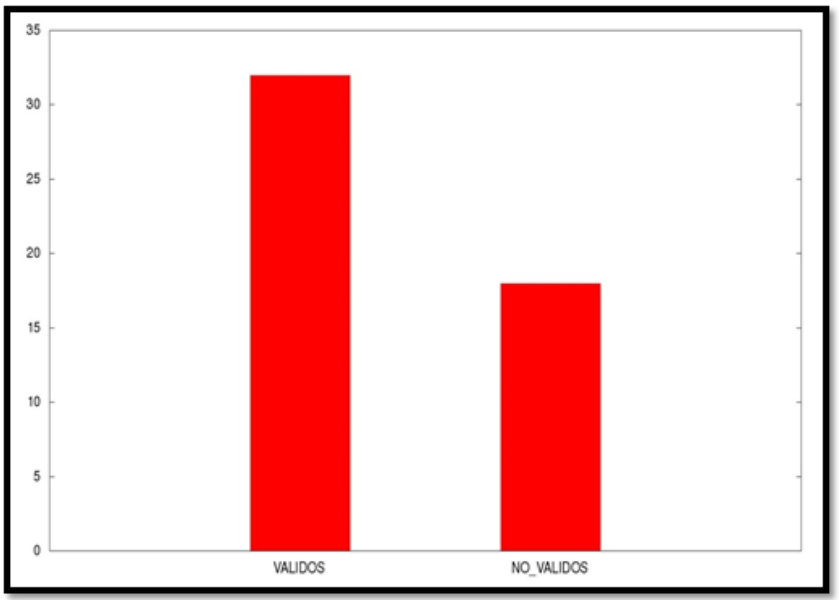


Ilustración 4. Gráfica de Validez

SALIDA: CATEGORIZACIÓN DE SUJETOS

Al igual que la gráfica anterior, este archivo muestra de manera visual la agrupación de los sujetos en tres grupos iguales. De esta manera, se agrupa la información en base a ciertos criterios, en este caso, el valor medio de SBBI de cada sujeto.

En el caso de que la división de sujetos no sea exacta, los sujetos excedentes se almacenarán en el Tercil central. En el siguiente ejemplo, se aprecia esto último perfectamente.

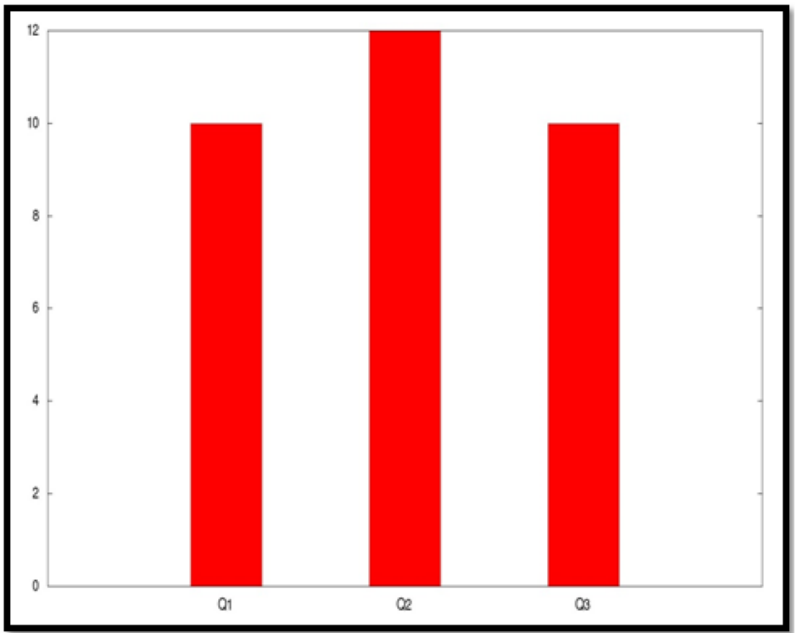


Ilustración 5. Gráfica de Categorización

SALIDA: ANÁLISIS DE SBBI

Esta gráfica muestra la agrupación de rupturas entre el comportamiento sedentario y la actividad física ligera de los días marcados como válidos. Los datos de esta gráfica se obtienen mediante contadores que se van incrementando durante la fase de análisis de la señal. Al encontrar una variación en el tipo de actividad y comprobar que se encuentra en una fase válida, se resetea el contador y se incrementa el número de rupturas.

Al observar estas graficas se puede observar una frecuencia de ruptura, la cual indica el número de veces que un sujeto pasa de actividad sedentaria a actividad ligera. Una gráfica con muchas rupturas indicará que el sujeto es muy activo (breaker) mientras que una frecuencia baja significa que el sujeto tiene un comportamiento con muy poca actividad física (prolonger). La siguiente grafica muestra una frecuencia de ruptura bastante alta, lo que significa que estamos ante un sujeto muy activo:

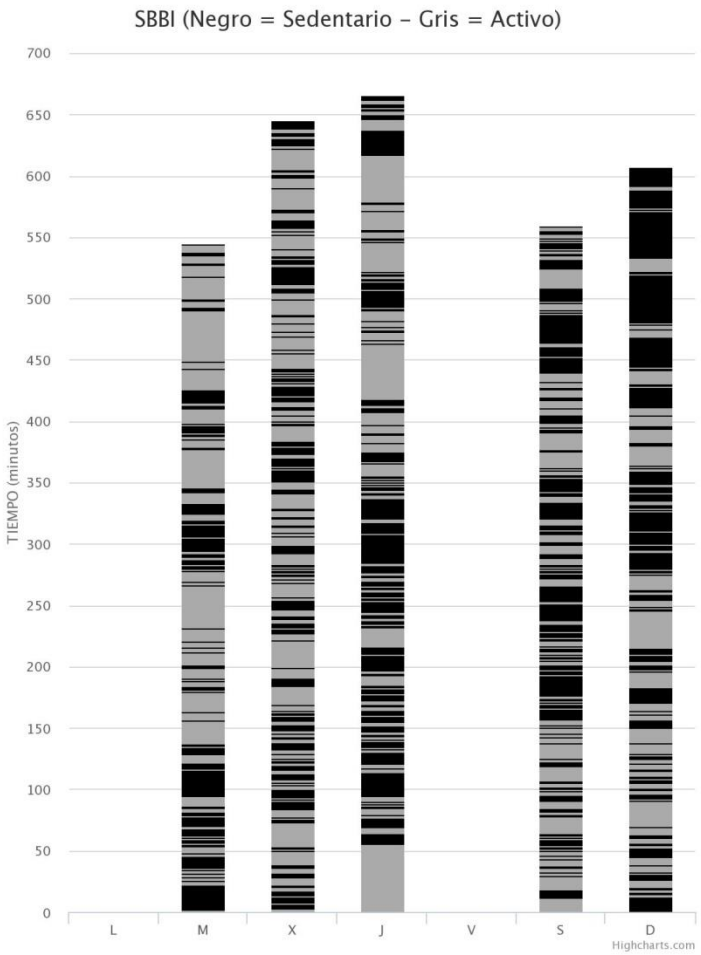
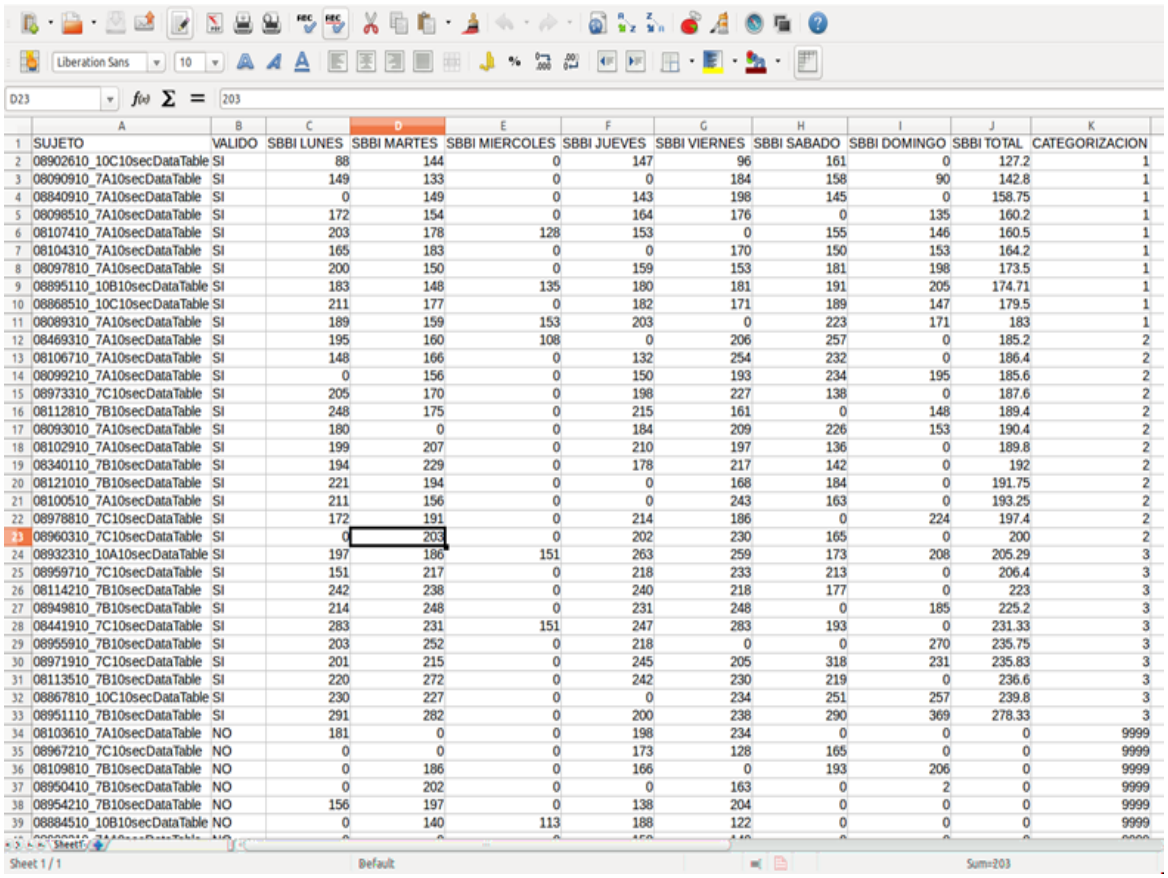


Ilustración 6. Gráfica de sujeto activo

SALIDA: ANALISIS DE SUJETOS

Por último, el programa genera un fichero que contiene una tabla con todos los sujetos analizados y ordenados por su SBBI y categorización. Esta tabla contiene varios campos: nombre del sujeto, que se obtiene del nombre del fichero, valor que indica si un sujeto es válido o no, el SBBI de cada día válido de la semana, la media de SBBI que se obtiene únicamente con el SBBI de los días válidos y la categorización de los sujetos en terciles. Un ejemplo de esta salida sería el siguiente:



	A	B	C	D	E	F	G	H	I	J	K
1	SUJETO	VALIDO	SBBI LUNES	SBBI MARTES	SBBI MIERCOLES	SBBI JUEVES	SBBI VIERNES	SBBI SABADO	SBBI DOMINGO	SBBI TOTAL	CATEGORIZACION
2	08902610_10C10secDataTable	SI	88	144	0	147	96	161	0	127.2	1
3	08090910_7A10secDataTable	SI	149	133	0	0	184	158	90	142.8	1
4	08840910_7A10secDataTable	SI	0	149	0	143	198	145	0	158.75	1
5	08098510_7A10secDataTable	SI	172	154	0	164	176	0	135	160.2	1
6	08107410_7A10secDataTable	SI	203	178	128	153	0	155	146	160.5	1
7	08104310_7A10secDataTable	SI	165	183	0	0	170	150	153	164.2	1
8	08097810_7A10secDataTable	SI	200	150	0	159	153	181	198	173.5	1
9	08895110_10B10secDataTable	SI	183	148	135	180	181	191	205	174.71	1
10	08868510_10C10secDataTable	SI	211	177	0	182	171	189	147	179.5	1
11	08089310_7A10secDataTable	SI	189	159	153	203	0	223	171	183	1
12	08469310_7A10secDataTable	SI	195	160	108	0	206	257	0	185.2	2
13	08106710_7A10secDataTable	SI	148	166	0	132	254	232	0	186.4	2
14	08099210_7A10secDataTable	SI	0	156	0	150	193	234	195	185.6	2
15	08973310_7C10secDataTable	SI	205	170	0	198	227	138	0	187.6	2
16	08112810_7B10secDataTable	SI	248	175	0	215	161	0	148	189.4	2
17	08093010_7A10secDataTable	SI	180	0	0	184	209	226	153	190.4	2
18	08102910_7A10secDataTable	SI	199	207	0	210	197	136	0	189.8	2
19	08340110_7B10secDataTable	SI	194	229	0	178	217	142	0	192	2
20	08121010_7B10secDataTable	SI	221	194	0	0	168	184	0	191.75	2
21	08100510_7A10secDataTable	SI	211	156	0	0	243	163	0	193.25	2
22	08978810_7C10secDataTable	SI	172	191	0	214	186	0	224	197.4	2
23	08960310_7C10secDataTable	SI	0	203	0	202	230	165	0	200	2
24	08932310_10A10secDataTable	SI	197	186	151	263	259	173	208	205.29	3
25	08959710_7C10secDataTable	SI	151	217	0	218	233	213	0	206.4	3
26	08114210_7B10secDataTable	SI	242	238	0	240	218	177	0	223	3
27	08949810_7B10secDataTable	SI	214	248	0	231	248	0	185	225.2	3
28	08441910_7C10secDataTable	SI	283	231	151	247	283	193	0	231.33	3
29	08955910_7B10secDataTable	SI	203	252	0	218	0	0	270	235.75	3
30	08971910_7C10secDataTable	SI	201	215	0	245	205	318	231	235.83	3
31	08113510_7B10secDataTable	SI	220	272	0	242	230	219	0	236.6	3
32	08867810_10C10secDataTable	SI	230	227	0	0	234	251	257	239.8	3
33	08951110_7B10secDataTable	SI	291	282	0	200	238	290	369	278.33	3
34	08103610_7A10secDataTable	NO	181	0	0	198	234	0	0	0	9999
35	08967210_7C10secDataTable	NO	0	0	0	173	128	165	0	0	9999
36	08109810_7B10secDataTable	NO	0	186	0	166	0	193	206	0	9999
37	08950410_7B10secDataTable	NO	0	202	0	0	163	0	2	0	9999
38	08954210_7B10secDataTable	NO	156	197	0	138	204	0	0	0	9999
39	08884510_10B10secDataTable	NO	0	140	113	188	122	0	0	0	9999

Ilustración 7. Fichero de categorización de sujetos

MANUAL DE INSTRUCCIONES

En los requisitos proporcionados se especifica la necesidad de crear un manual de usuario. Este documento se encuentra en el anexo A junto con una serie de capturas para facilitar la navegabilidad por la aplicación.

5.3. EJECUCIÓN DETALLADA

Una vez mostrado el diseño de las entradas y salidas y componentes de la aplicación, se mostrará el funcionamiento de la aplicación de una manera detallada dividido en tres secciones.

PRE-PROCESADO

Una vez ejecutado el programa, se muestra el siguiente menú

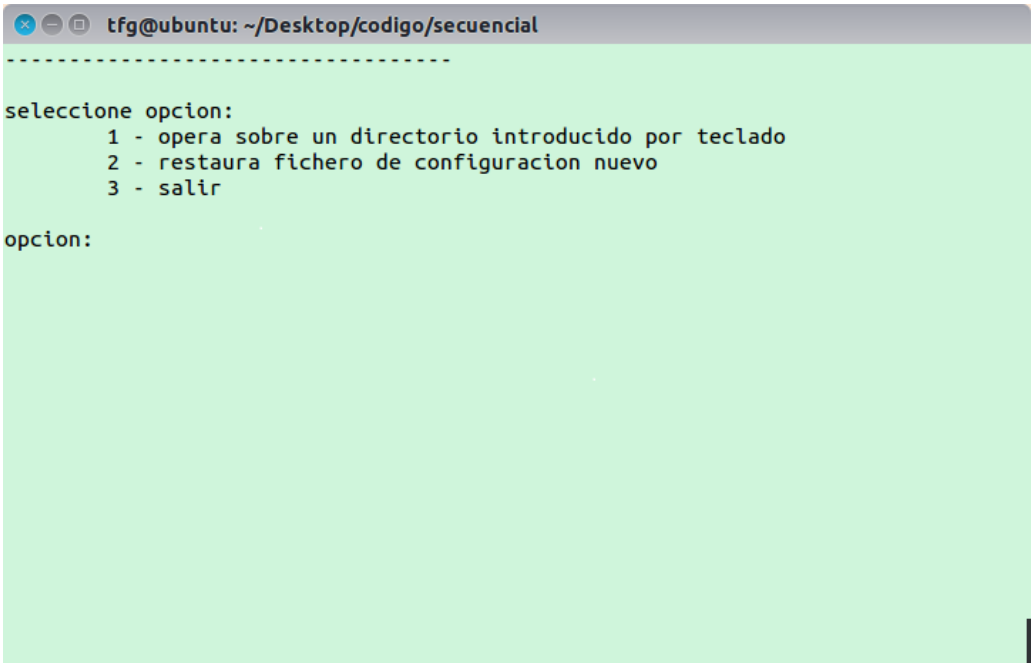


Ilustración 8. Menú principal

Para este apartado ejecutaremos la opción 1, e introduciremos un directorio con 10 sujetos para analizarlos. El programa se posicionará en ese directorio y recorrerá los archivos del directorio comprobando los que tienen el formato acordado y copiando su nombre en una lista y los que no. La lista de ficheros se utilizará después, mientras que el número de ficheros válidos y no válidos servirá para controlar errores y mostrar información al usuario.

Una vez tenemos la lista de ficheros que vamos a analizar, se carga la configuración como se explica antes y se crea el fichero de resultados con su cabecera.

PROCESADO

El programa entra en un bucle utilizando el número de sujetos válidos como número de iteraciones a realizar. Por cada iteración, se carga en memoria un sujeto de la lista obtenida anteriormente y se filtra su señal segundo a segundo utilizando los parámetros de configuración. Como el filtrado es destructivo, se necesita una lista auxilair donde ir guardando los valores válidos.

Para el indexado de las listas (original y auxiliar) se utilizan dos índices que van recorriendo la lista original. El primero sirve como guarda para marcar el inicio de un intervalo mientras que el segundo se va incrementando si se cumple la condición de tiempo no vestido o movimiento artificial. Al dejar de cumplirse, se comprueba la longitud del intervalo y se decide

si se elimina o no. En el primer caso, se modifica el índice inicial para que apunte al índice final mientras que en el segundo caso, se copia en la lista auxiliar todo el intervalo.

Una vez filtrado, se marca al sujeto como válido o no válido. En caso de ser no válido, se inserta en una lista de sujetos no válidos que se usará en el post procesado. En caso de ser válido, se calcula su actividad física junto con el SBBI y se genera el fichero HTML con la gráfica asociada a ese sujeto. Por último, se inserta en una lista de sujetos válidos que se usará más tarde. En cualquier caso, se libera la memoria del sujeto leído para evitar sobrecargas y se imprimirá el resultado por pantalla.

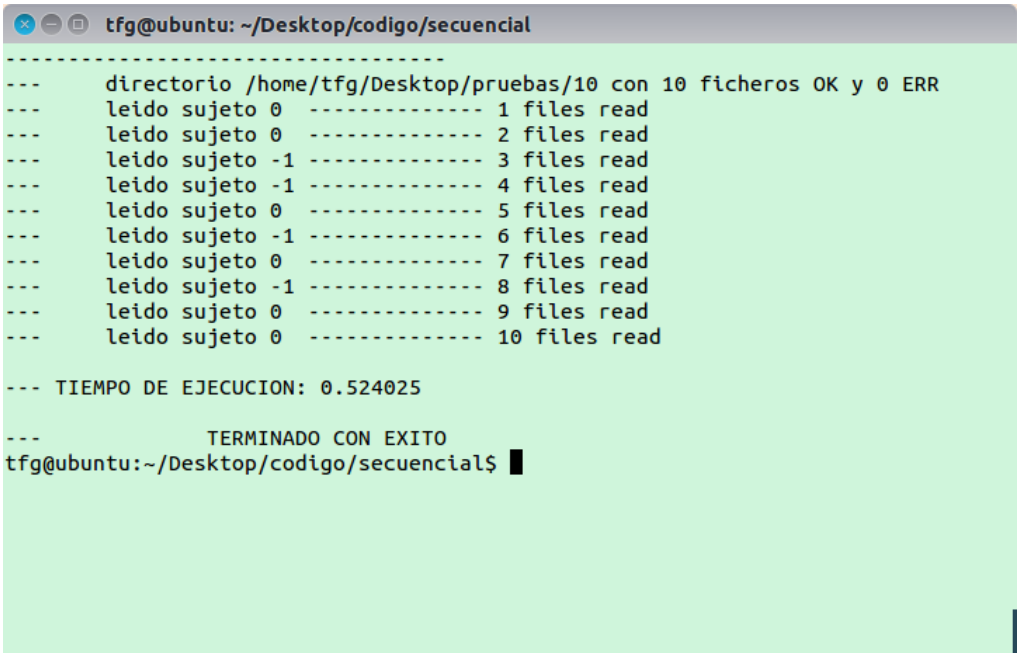


Ilustración 9. Ejecución finalizada

POST-PROCESADO

Al finalizar el bucle, se calcula la categorización de los sujetos válidos y se ordenan. Este proceso se realiza fuera del bucle, ya que se depende del resultado de todos los sujetos para poder realizarse. Una vez ordenados y categorizados, se procede a su impresión en el fichero de resultados. A continuación se imprimen también los sujetos no válidos y se cierra el fichero.

Por último, se generan las gráficas de la categorización y de participación en el estudio y se liberan las listas de sujetos válidos y no válidos y la lista de ficheros leída en el pre-procesado.

5.4. PARALELISMO

La aplicación tiene la propiedad de poder ejecutarse de manera paralela para mejorar el rendimiento y reducir el tiempo de ejecución, ya que el número de sujetos a analizar es muy elevado.

CLASIFICACIÓN Y MODELO DE PROGRAMACIÓN

La arquitectura paralela utilizada sigue el modelo MIMD dentro de la clasificación de Flynn[6], más concretamente en su variante con memoria común. Esta arquitectura tiene la ventaja de utilizar memoria compartida, ya que se basa en el uso de un multiprocesador en lugar de un sistema multicomputador.

Con esta arquitectura de memoria compartida, se propone una paralelización de grado fino, ya que se implementa a nivel de bucles y sentencias. Este modelo de programación permite que varios procesos se ejecuten en un espacio de direcciones común, por lo que la comunicación entre éstos está implícita en el acceso a memoria. Como desventajas de este modelo, destaca la poca escalabilidad (O (16) procesadores) y el alto coste del hardware.

Para llevar a cabo la paralelización del bucle, se han analizado las dependencias para evitar sobrescribir variables y alterar los valores de los sujetos. En la aplicación se ha encontrado una dependencia de tipo RAW (Read-After-Writting)[7]. Esta dependencia hace referencia a una situación donde se necesita un dato que aún no ha sido calculado, por lo que la instrucción implicada no puede ser paralelizada y se ejecuta de manera secuencial.

Una vez solventadas las dependencias, se ha procedido a implementar el reparto de iteraciones entre los procesadores. Para ello, se ha seguido una planificación dinámica en la que cada procesador coge una nueva iteración según acaba la anterior. Con esto se consigue un equilibrio de carga, ya que no quedan procesadores ociosos mientras otros terminan sus iteraciones. Este reparto dinámico puede añadir una carga adicional, ya que recoge más iteraciones que una distribución fija, pero con la relación al coste de las tareas que se asignan, esta carga es despreciable.

El diagrama de ejecución de la aplicación una vez aplicada la paralelización del bucle con las dependencias resueltas es el siguiente:

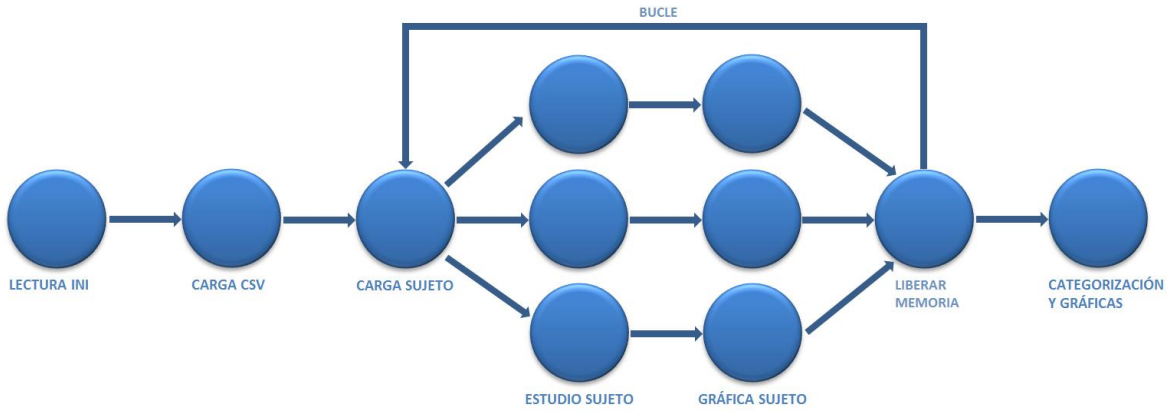


Ilustración 10. Diagrama de ejecución

Como se aprecia, existen partes secuenciales antes del bucle principal. Estas partes no se pueden paralelizar ya que están formadas por instrucciones que deben ser ejecutadas secuencialmente y por el menú de la aplicación.

Antes del bucle se ha definido una norma que se debe seguir para el reparto de los sujetos entre los procesadores. Para el estudio posterior se utilizará un reparto de un sujeto por cada procesador y después se comparará con un estudio realizado para un reparto de 10 sujetos por procesador.

Dentro del bucle, se ha solucionado el problema de la dependencia RAW de la lectura de sujetos mediante una ejecución secuencial de la instrucción. Esta ejecución se realiza mediante una sección crítica para evitar que se ejecute por más de un proceso a la vez.

Tras finalizar el bucle, existen otras instrucciones secuenciales que no pueden ser paralelizadas. Estas instrucciones están formadas por la categorización de sujetos y la impresión en fichero, ya que la primera sufre otra dependencia RAW y la segunda solo permite la escritura en fichero por un proceso simultaneo para evitar el solapamiento de datos.

RENDIMIENTO

Una vez definida la parte secuencial y paralela de la aplicación, se puede comenzar a evaluar el rendimiento total. La primera medición recoge los tiempos de ejecución. Se han seleccionado grupos de sujetos aleatorios de distintos tamaños (10, 20, 50, 100, 200, 400, 700 y 1000) para poder realizar una curva con los tiempos del programa secuencial y con 2, 3 y 4 procesadores (las tablas con los valores obtenidos se encuentran en el anexo B).

Ilustración 11. Gráfica de tiempos de ejecución

Como se puede apreciar, el rendimiento de los programas paralelos respecto al tiempo en serie no mejora mucho. Esto se debe a que la parte paralela solo se ejecuta al marcar un sujeto como válido. En caso de encontrar sujetos no válidos, el programa sigue una ejecución similar a la secuencial. De esto podemos concluir que a más sujetos válidos, mejor rendimiento se obtendrá, como se explicará más adelante.

Para poder medir la mejora del rendimiento observado definimos una función de aceleración (speed-up) con la siguiente formula:

$$SpeedUp = \frac{Timepo\ Secuencial}{Tiempo\ Pparalelo}$$

Cuanto mayor sea la aceleración, mejor rendimiento tendrá. El problema en este caso es que la aceleración está acotada superiormente por la parte secuencial y la aparición de sujetos no válidos que no pueden aprovechar la paralelización. De esta manera se limita la ganancia de velocidad de manera importante[8].

Ilustración 12. Gráfica del Speed-Up respecto al tiempo

Como se ve en la imagen, la aceleración con 4 procesadores es algo mayor que con el resto. Esto quiere decir que a mayor cantidad de procesadores, mayor aceleración tendrá la aplicación. Este dato se ve demostrado cuando obtenemos el speed-up completo al analizar los 1000 sujetos.

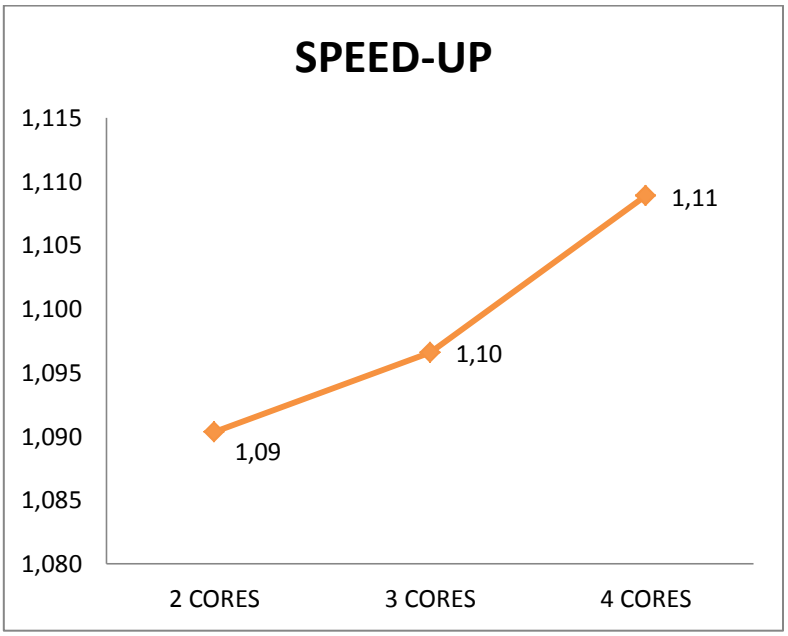


Ilustración 13. Speed-Up total

Además del speed-up se han medido otros parámetros que se han considerado importantes a la hora de medir rendimientos como el overhead, la eficiencia (aceleración entre procesadores) o el coste (número de procesadores por tiempo).

El overhead de la aplicación es la suma del tiempo de esperas, exceso de computación, comunicaciones... que se llevan a cabo para la sincronización y acceso a memoria por los procesadores indicados mediante las directivas OpenMP. Se define mediante la siguiente función:

$$Overhead = procesadores * Tiempo\ Paralelo - Tiempo\ Secuencial$$

Como se puede ver en la primera gráfica, a más procesadores, mayor es el tiempo de overhead. Esto tiene sentido ya que se necesita más tiempo para poder sincronizar todos los procesadores y requieren de más tiempo de espera en la cola para el acceso a memoria. Esto ralentiza el programa ya que el trabajo realizado por los procesadores es muy pequeño y se realizan muchas llamadas a la memoria compartida. En la segunda gráfica se ve como el overhead crece de manera exponencial según se analizan los sujetos, demostrando otra vez que a mayor número de procesadores, mayor él es tiempo de overhead.

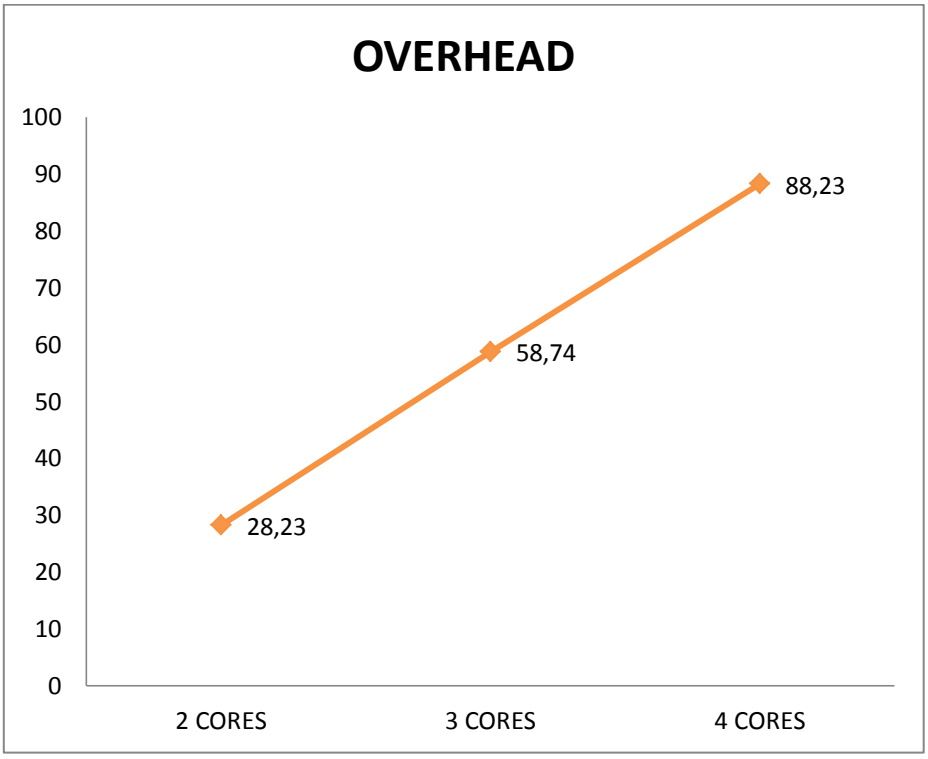


Ilustración 14. Gráfica del overhead total

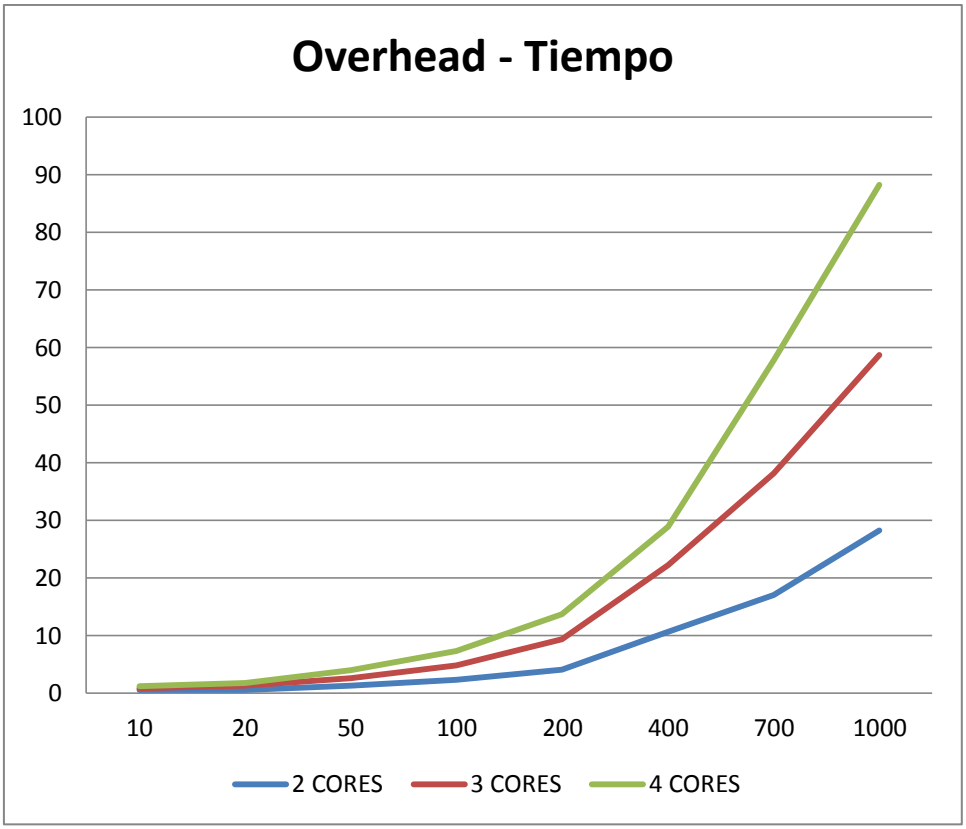


Ilustración 15. Gráfica del Overhead respecto al tiempo

La eficiencia de la aplicación viene determinada por el speed-up y el número de procesadores implicados en esa aceleración mediante la siguiente función:

$$Eficiencia = \frac{SpeedUp}{procesadores}$$

Si el resultado tiende a 0, querrá decir que el paralelismo es poco eficiente y el resultado es cercano al tiempo secuencial (lo cual se puede intuir viendo los resultados anteriores). En cambio, si el resultado tiende a 1, querrá decir que todos los procesadores están trabajando en todo momento.

Como se aprecia, la eficiencia va disminuyendo a medida que se utilizan más procesadores. Esto se produce por el incremento del tiempo ocasionado por el overhead.

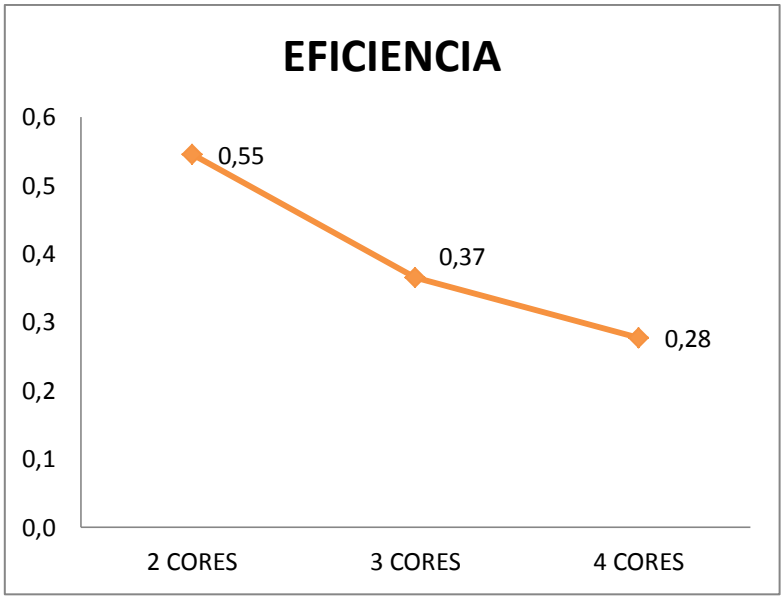


Ilustración 16. Gráfica de la eficiencia

Los costes indican la suma de tiempo que cada elemento de proceso dedica en la resolución del problema. Este valor se obtiene mediante la fórmula

$$Coste = procesadores * Tparalelo$$

El coste del paralelismo será óptimo si el coste del programa paralelo es idéntico al coste del secuencial. En este caso, como la mejora del tiempo no es muy efectiva, se produce un aumento considerable del coste. Este coste aumenta con el número de procesadores ya que los tiempos obtenidos para los distintos números de procesadores son muy similares.

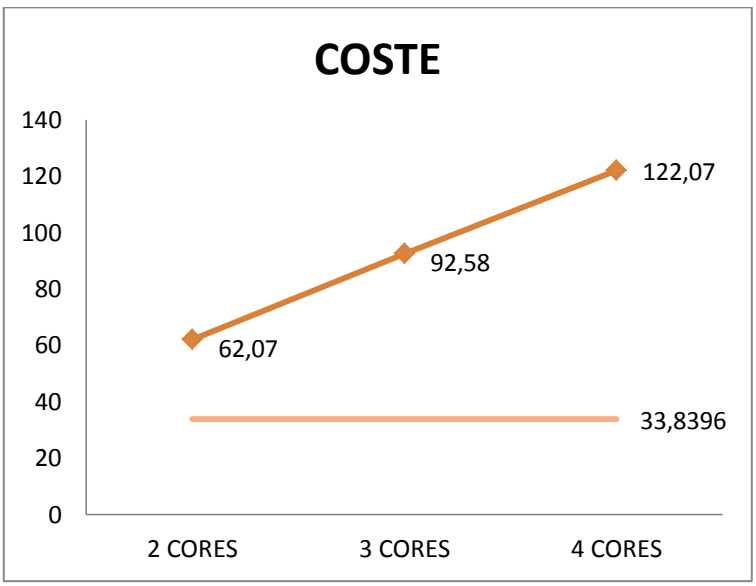


Ilustración 17. Gráfica de costes

Para poder determinar si es posible mejorar la aplicación y obtener un mejor rendimiento, se ha modificado el sistema de reparto de los sujetos para cada procesador. En las gráficas anteriores se utilizaba un reparto de un sujeto por cada procesador, lo que ocasionaba un aumento del overhead y una disminución de la eficiencia.

Para este nuevo estudio se utiliza un reparto de 10 sujetos por procesador con la intención de evitar procesadores ociosos al recibir sujetos no válidos, reducir el tiempo de acceso a memoria y aumentar el tiempo de ejecución en cada procesador (las tablas con los datos se encuentran en el anexo C).

Los tiempos obtenidos con esta nueva configuración se asemejan mucho con los obtenidos al principio, pero en el resto de gráficas si se observa una pequeña mejora de rendimiento

Ilustración 18. Gráfica de tiempos con distribución de sujetos igual a 10

En esta gráfica se observa la diferencia entre el speed-up de la ejecución con distribución igual a 10 y distribución igual a 1. El speed-up mejora con la nueva distribución respecto a la anterior, pero esta mejoría va disminuyendo con el número de procesadores debido al coste del overhead.

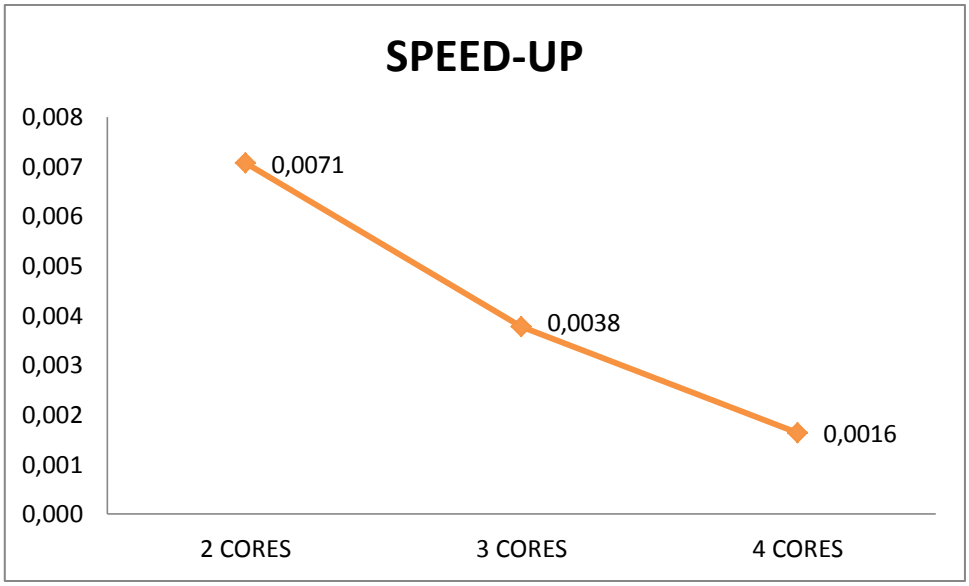


Ilustración 19. Diferencias de Speed-Up

En el caso del overhead, se reduce el tiempo indirecto de ejecución ya que cada procesador hace menos peticiones a la memoria compartida. Usando 2 procesadores se consigue una mejora que se va reduciendo a la vez que aumentamos el número de procesadores. Esto se produce porque el overhead del nuevo método de reparto tiende al método anterior ya que a más procesadores, más alto será el valor esperado. El overhead de la nueva distribución no será nunca mayor que la anterior ya que se realizan menos llamadas a la memoria compartida.

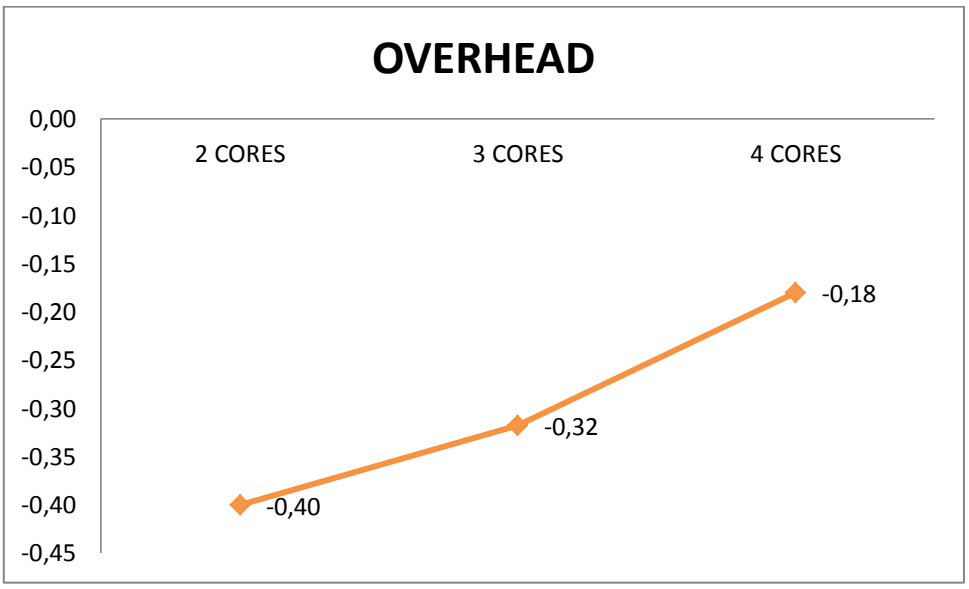


Ilustración 20. Diferencias de Overhead

La eficiencia tiene un comportamiento similar al speed-up. El valor se va reduciendo con el número de procesadores ya que el tiempo de ejecución no se reduce tanto como se esperaba pese a reducir el overhead.

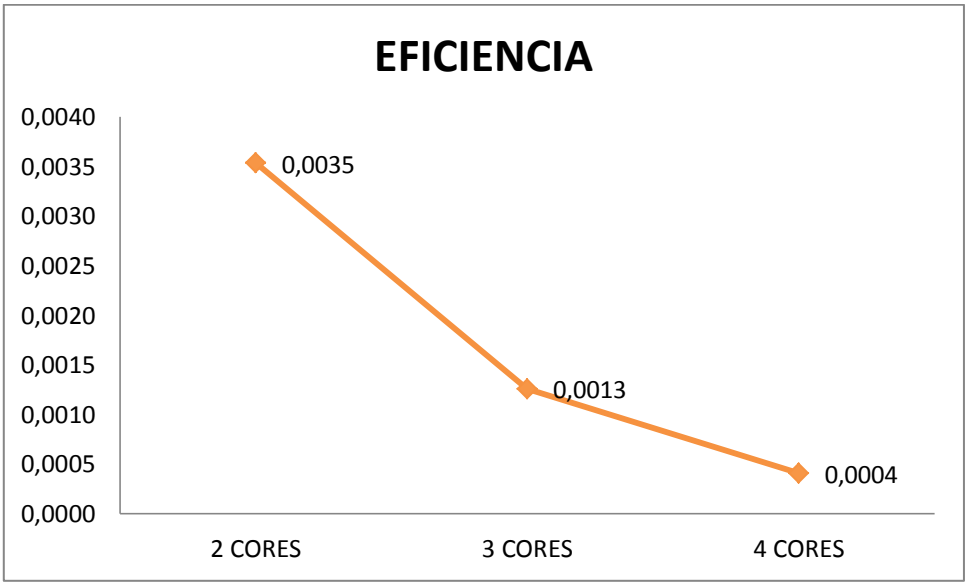


Ilustración 21. Diferencias de eficiencia

Los costes tienen un comportamiento muy parecido al overhead. Al reducir un poco los tiempos de ejecución, se reducen los costes. Esta mejora se va reduciendo ya que los tiempos empiezan a converger según aumenta el número de sujetos analizados.

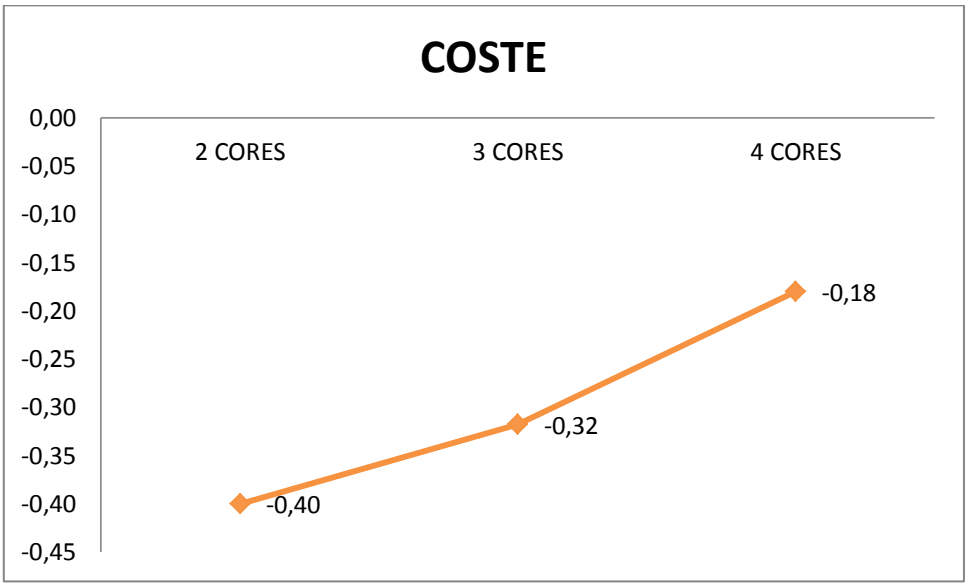


Ilustración 22. Diferencias de costes

OTRAS PRUEBAS REALIZADAS

Para comprobar que realmente se puede mejorar el rendimiento de la aplicación, se hace una última prueba utilizando solamente ficheros se sujetos válidos. Con esto se pretende ver el alcance máximo de la aplicación, ya que en este supuesto extremo, se aprovecharía totalmente el paralelismo.

Para esta nueva ejecución se ha buscado un grupo de sujetos válidos y se han copiado varias veces para coincidir con el número de pruebas realizadas anteriormente y poder comparar los resultados. El reparto que llevan a cabo los procesadores es de 10 sujetos por procesador, ya que los resultados obtenidos son algo mejores que utilizando un reparto de un solo sujeto por procesador (las tablas con los valores obtenidos se encuentran en el anexo D).

Los tiempos obtenidos para la primera gráfica son algo mayores que en la ilustración 9 ya que se tienen que procesar todos los datos y no se desechan sujetos no válidos. También se aprecia una mayor separación entre el tiempo secuencial y los tiempos en paralelo. Esta separación se ve mejor en la segunda gráfica, que muestra la diferencia entre los tiempos de cada ejecución.

Ilustración 23. Tiempos de ejecución con todos los sujetos válidos

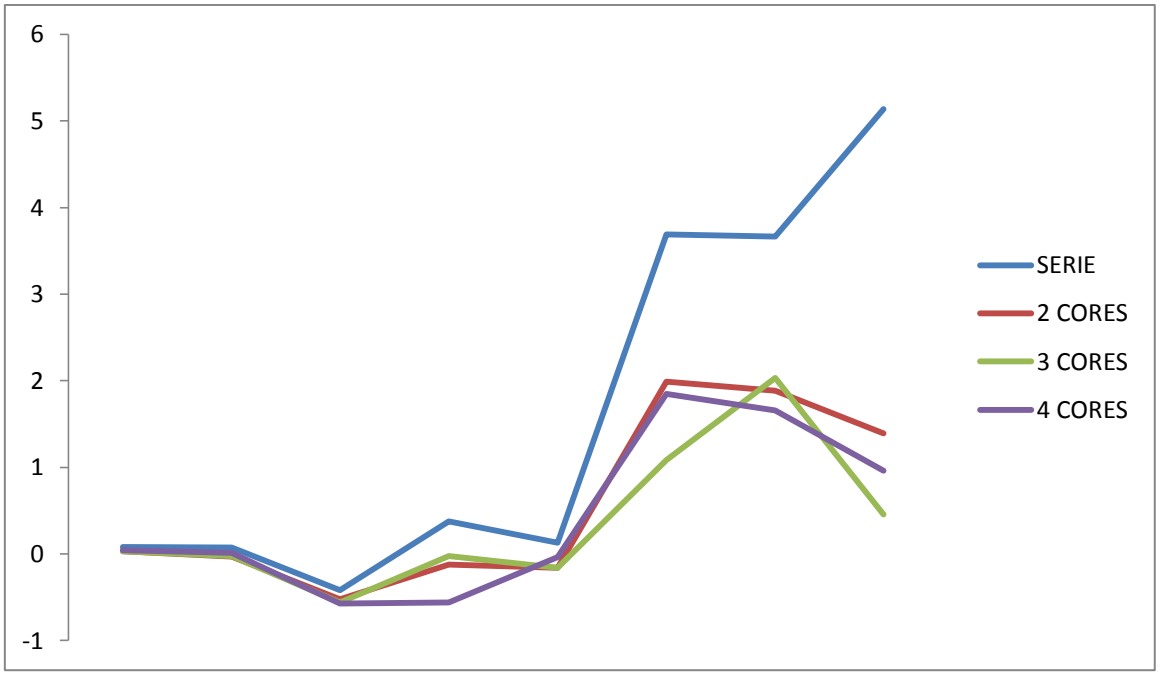


Ilustración 24. Diferencia de tiempos entre los sujetos válidos y los originales

Al existir más diferencia, podemos deducir que la aplicación en paralelo es más óptima cuanto mayor es el número de sujetos válidos. Por tanto, nos interesa obtener cuanta es la mejora máxima que permite esta aplicación.

El primer parámetro que se calcula es el speed-up máximo que puede alcanzar el programa. Como se ve en la siguiente gráfica, el nuevo valor obtenido mediante la resta de los speed-ups de las ejecuciones originales y las propuestas, indica que se puede mejorar el rendimiento en un 0,13% de media.

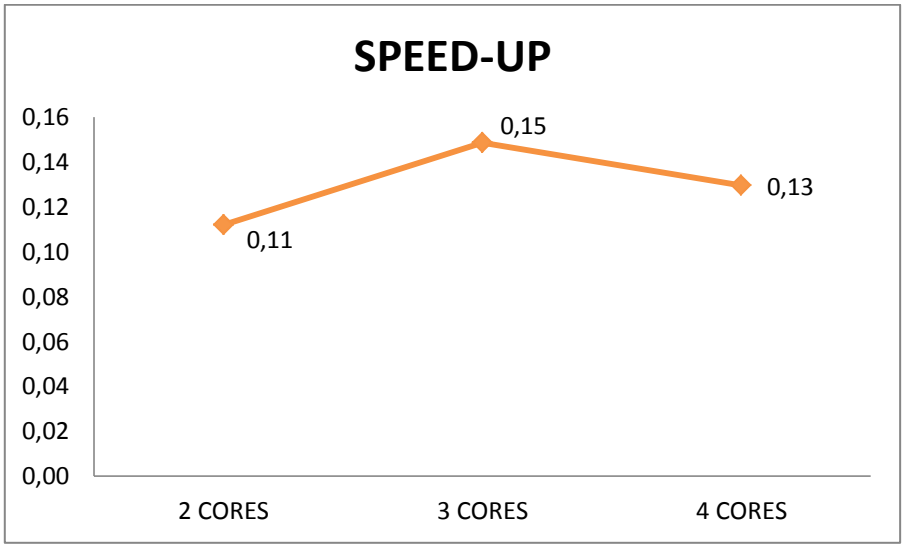


Ilustración 25. Diferencia de Speed-Ups

Ocurre lo mismo con el overhead, cada procesador está ocupado más tiempo, por lo que el tiempo de acceso a la memoria compartida disminuye respecto al tiempo total en unos 2 segundos de media.

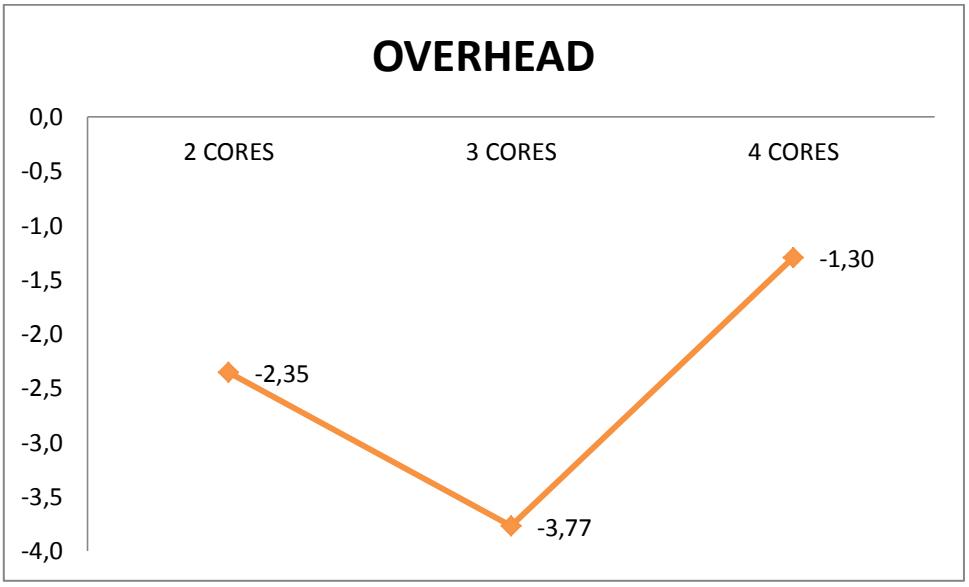


Ilustración 26. Diferencia de Overhead

La eficiencia también mejora, ya que los procesadores están más tiempo ocupados. Este aumento, al igual que le ocurría a la eficiencia en los repartos de un sujeto y de 10, disminuye con el número de procesadores debido al tiempo de overhead.

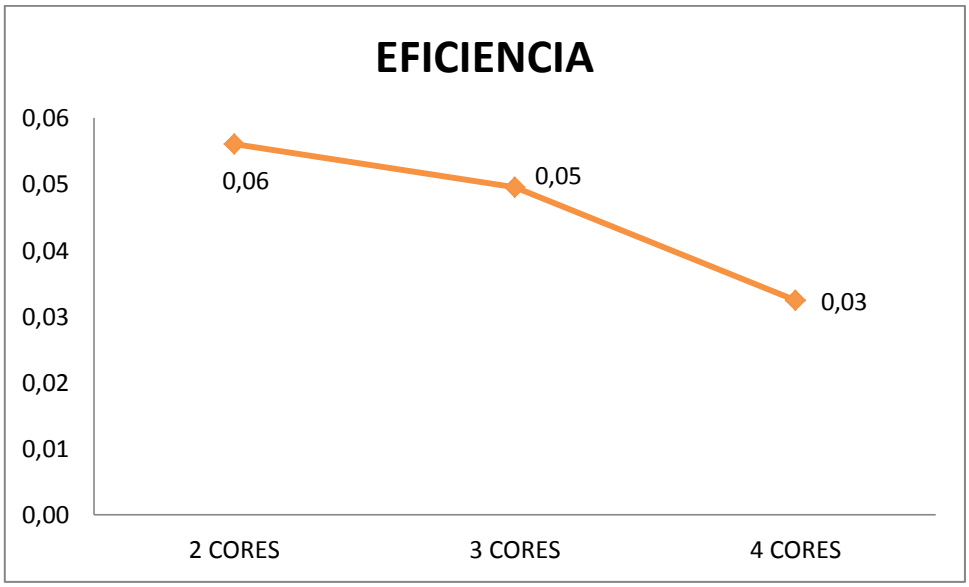


Ilustración 27. Diferencia de eficiencias

Al estar más tiempo procesando los sujetos, el coste crece respecto a los valores originales ya que el uso de los procesadores aumenta. El aumento es poco significativo, apenas un 4% del coste original.

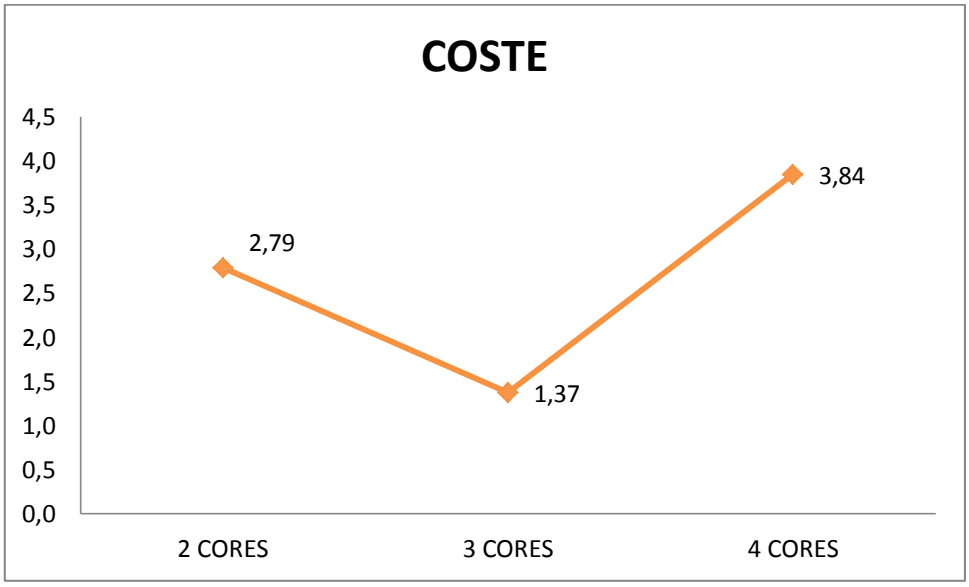


Ilustración 28. Diferencia de costes

6. CONCLUSIONES

En este proyecto se ha desarrollado una aplicación que permite analizar sujetos y generar una serie de gráficas tras filtrar sus señales. Este trabajo se ha desarrollado de tal manera que facilite la incorporación de nuevos procedimientos de filtrado o análisis en estudios posteriores.

También se ha realizado un estudio e implementación de paralelismo de datos para poder optimizar los tiempos de ejecución obtenidos y poder realizar más trabajo sin necesidad de esperar al programa.

El proyecto consta de dos sub proyectos, uno ya paralelizado y otro totalmente secuencial. El motivo es poder facilitar la implementación paralela mediante otro sistema distinto de OpenMP (CUDA, MPI...) y que no exista necesidad de programar de nuevo todo el sistema de filtrado y generación de datos.

Los datos obtenidos en este proyecto forman parte de un estudio realizado por los compañeros de la facultad de magisterio que busca encontrar una manera para poder generar una gráfica con el índice de ruptura del comportamiento sedentario, ya que los programas que utilizan no recogen ese parámetro.

Mientras se realizaban las pruebas de rendimiento se han encontrado diversas limitaciones debido a que el número de ficheros a analizar es reducido y a nivel de usuario la mejoría en rendimiento va a ser poco notable.

TRABAJO FUTURO

En un futuro se podrían estudiar varios temas para aumentar el alcance del proyecto. Por ejemplo, se podrían añadir nuevas formas de filtrado de los sujetos, obtener nuevos datos relacionando el SBBI obtenido con algún otro parámetro del acelerómetro, exportar la aplicación a entornos Android e iOS o nuevas formas de paralelismo.

7. REFERENCIAS

[1] http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3184184/table/T2/

[2] http://programador-apli.blogspot.com.es/2012/04/calcular-el-dia-de-la-semana-partir-de.html

[3] http://www.highcharts.com/products/highcharts

[4] http://gnuplot.sourceforge.net/demo/histograms.html

[5] http://gnuplot.sourceforge.net/demo/histograms.html

[6] http://es.wikipedia.org/wiki/Taxonom%C3%ADa_de_Flynn

[7] http://en.wikipedia.org/wiki/Hazard_(computer_architecture)

[8] http://es.wikipedia.org/wiki/Ley_de_Amdahl

8. GLOSARIO

OpenMP: interfaz de programación de aplicaciones para la programación multiproceso de memoria compartida en múltiples plataformas.

SBBI: término utilizado por el equipo de investigación para referirse a índice de ruptura del comportamiento sedentario (Sedentary Behavior Break Index) o a la media de tiempo que tarda un sujeto en cambiar de comportamiento sedentario a actividad física ligera o viceversa.

Terciles: puntos tomados a intervalos regulares para dividir la distribución de sujetos en tres partes iguales.

Epoch: intervalo de tiempo entre mediciones del acelerómetro.

TAD: siglas de Tipos Abstractos de Datos. Los TADs son modelos matemáticos para tipos de datos donde estos tipos son definidos por su comportamiento desde un punto de vista del usuario.

CPM: counts per minute. Este término se refiere a la agrupación de counts por minuto. Se utiliza para el análisis de la actividad física de los sujetos.

Breaker: sujeto con un índice de ruptura del SBBI alto.

Prolonger: sujeto con un SBBI bajo.

Speed-Up: valor que indica la mejora del tiempo paralelo respecto al tiempo secuencial mediante la división de estos valores.

Overhead: exceso de computación indirecta (acceso a memoria, ancho de banda).

9. ANEXOS

ANEXO A. Manual de instrucciones de la aplicación.

En el siguiente apartado se mostrará cómo generar la aplicación, ejecutarla y como interpretar las salidas y exportarlas al formato deseado, así como la navegabilidad de la aplicación.

COMPILACION Y EJECUCION

Para compilar el programa, hay que abrir un terminal y posicionarse en la carpeta del proyecto (secuencial o paralelo) y llamar al comando "make". Este comando automáticamente recogerá las dependencias dentro del código y generará los objetos que se enlazarán en el programa principal.

```
tfg@ubuntu: ~/Desktop/codigo/secuencial
                LIMPIANDO
rm -fr plot2.jpg plot3.jpg RESULTADOS.csv
rm -fr graficas/*.html obj/*.o mainSecuencial
rm -fr *.tmp *.gp
                COMPILANDO srclib/dictionary.c
gcc -Wall -I./include -o obj/dictionary.o -c srclib/dictionary.c
                COMPILANDO srclib/estudio.c
gcc -Wall -I./include -o obj/estudio.o -c srclib/estudio.c
                COMPILANDO srclib/filtrado.c
gcc -Wall -I./include -o obj/filtrado.o -c srclib/filtrado.c
                COMPILANDO srclib/iniparser.c
gcc -Wall -I./include -o obj/iniparser.o -c srclib/iniparser.c
                COMPILANDO srclib/manejoFicheros.c
gcc -Wall -I./include -o obj/manejoFicheros.o -c srclib/manejoFicheros.c
                COMPILANDO srclib/salida.c
gcc -Wall -I./include -o obj/salida.o -c srclib/salida.c
                COMPILANDO srclib/sujeto.c
gcc -Wall -I./include -o obj/sujeto.o -c srclib/sujeto.c
                COMPILANDO src/mainSecuencial.c
gcc -Wall -I./include -o obj/mainSecuencial.o -c src/mainSecuencial.c
gcc -Wall -o mainSecuencial obj/dictionary.o obj/estudio.o obj/filtrado.o obj/in
iparser.o obj/manejoFicheros.o obj/salida.o obj/sujeto.o obj/mainSecuencial.o
                EJECUTANDO
./mainSecuencial
```

Ilustración 29. Llamada a la regla make.

Una vez compilado, el fichero makefile ejecutará el programa (en el caso de ejecutar el proyecto paralelo, se ejecutará con 2 procesadores).

Existe otra posible llamada al ejecutable mediante la cual se le pueden especificar por argumentos algunos parámetros y así saltarse el menú principal.

En el caso del proyecto secuencial, el programa admite un argumento indicando la ruta en la que se encuentran los ficheros, mientras que en secuencial se admite el número de procesadores a utilizar o el número de procesadores y la ruta de los ficheros. En cualquiera de estos casos, se salta el menú principal y comienza el análisis de los sujetos.

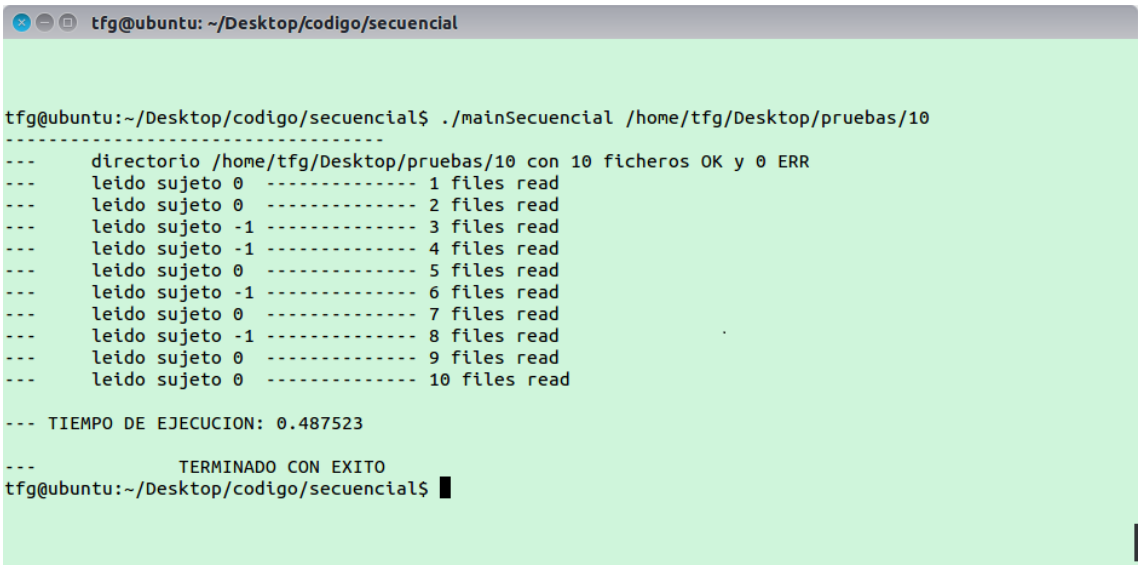


Ilustración 30. Llamada con argumentos al programa secuencial.

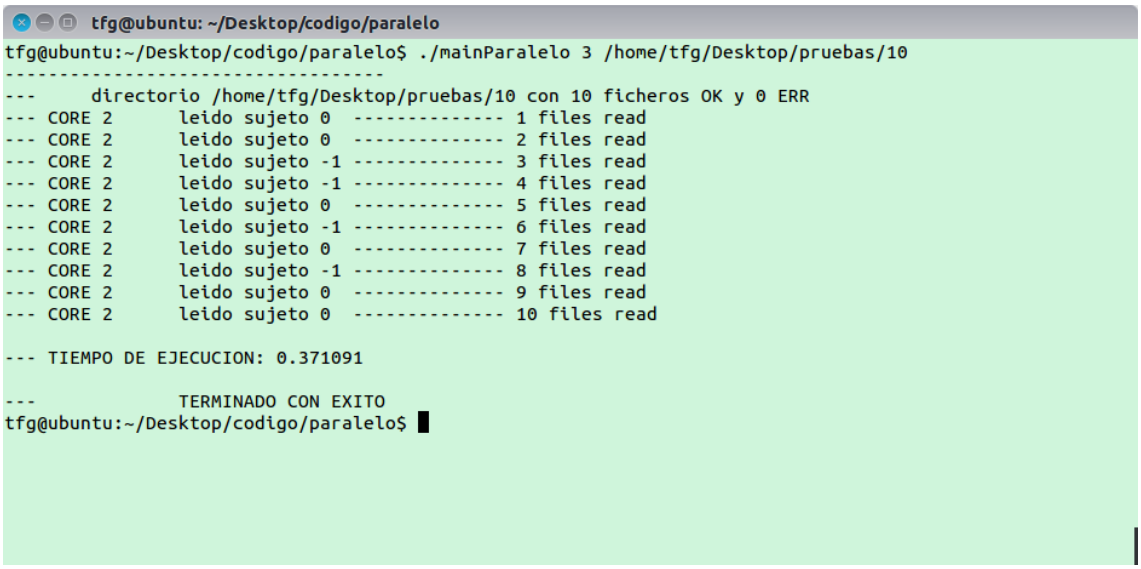


Ilustración 31. Llamada con argumentos al programa paralelo.

INTERFAZ DE USUARIO

Si el programa se ejecuta sin argumentos, aparecerá el siguiente menú en el que se podrá seleccionar alguna de las opciones disponibles:

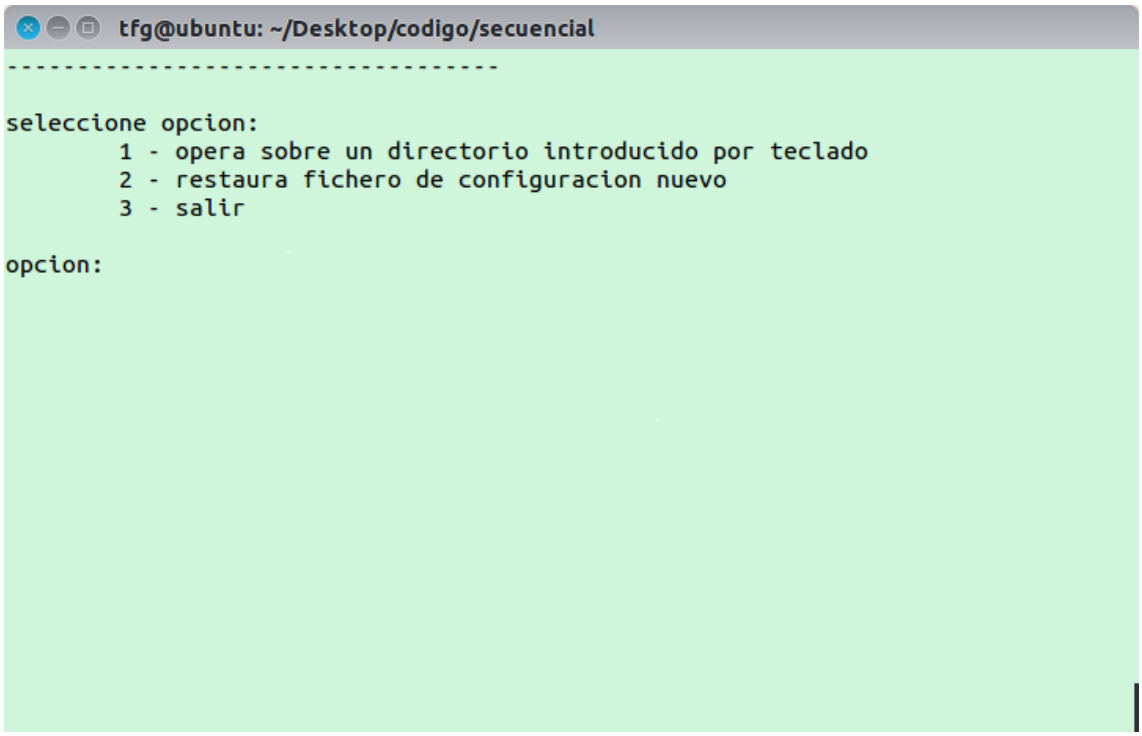


Ilustración 32. Menú de usuario.

Ejecución sobre un directorio: cuando se selecciona esta opción, se pide a continuación la ruta en la que el programa buscará los ficheros a analizar. En caso de introducir una ruta incorrecta, se volverá al menú principal.

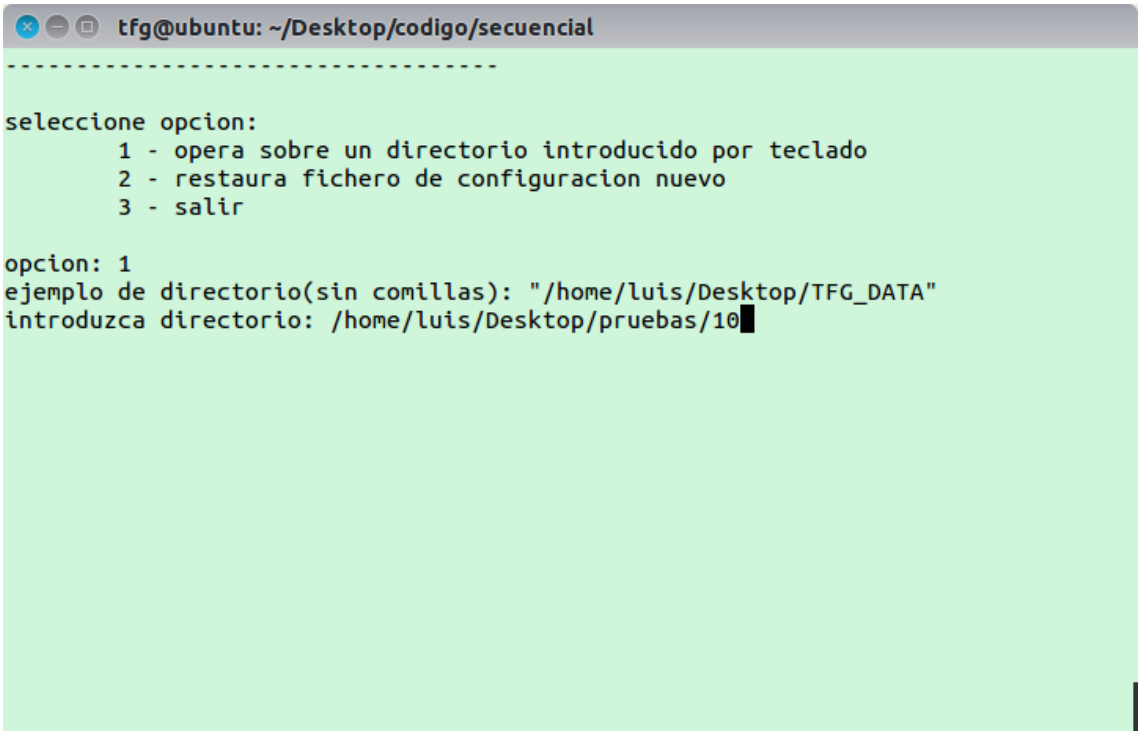


Ilustración 33. Introducción del directorio a analizar.

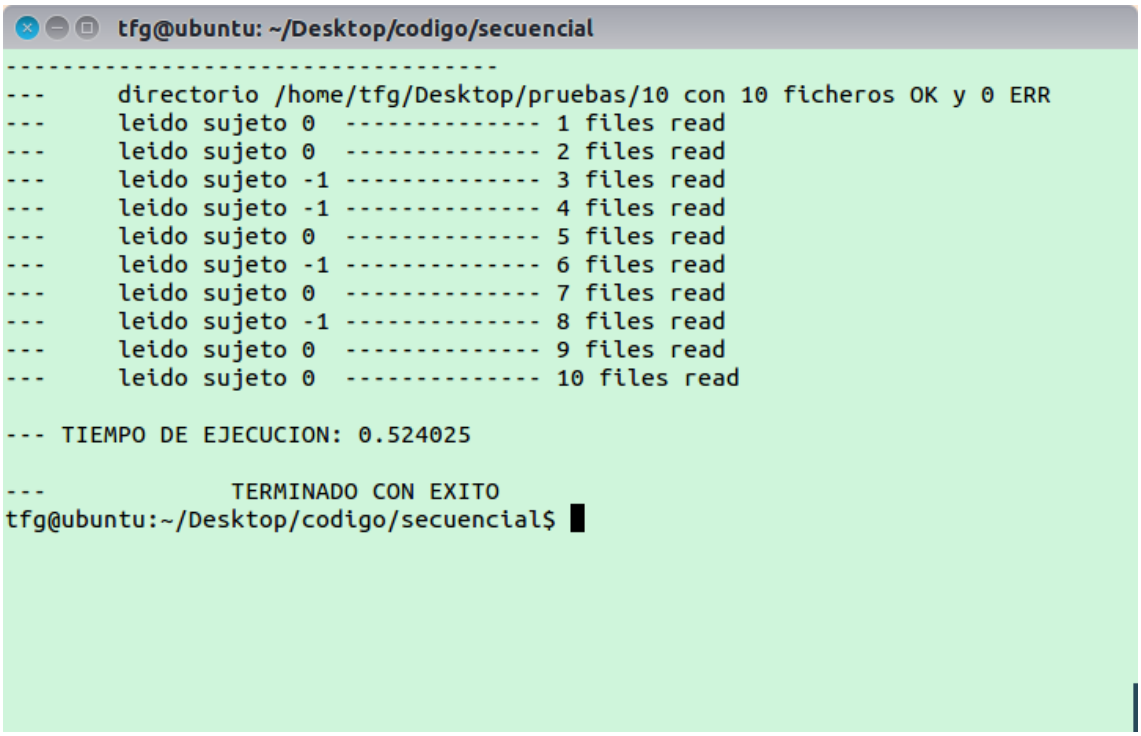


Ilustración 34. Resultado de la ejecución anterior.

Restaurar fichero de configuración: esta opción elimina el fichero de configuración actual y genera uno completamente nuevo. Esta opción evita que la aplicación deje de ser útil en caso de pérdida o mala gestión de la configuración.

Salir: al marcar esta opción se sale de la aplicación.

FICHEROS DE SALIDA

Los ficheros de salida se generan dentro de la carpeta del proyecto y en la carpeta "GRAFICAS". Los ficheros que contienen el SBBI se obtienen mediante una librería que genera una página web. Para poder exportar este archivo a una imagen PNG, JPEG o incluso a PDF, se debe abrir el fichero y hacer clic en el icono de la esquina superior derecha para abrir el menú de opciones.

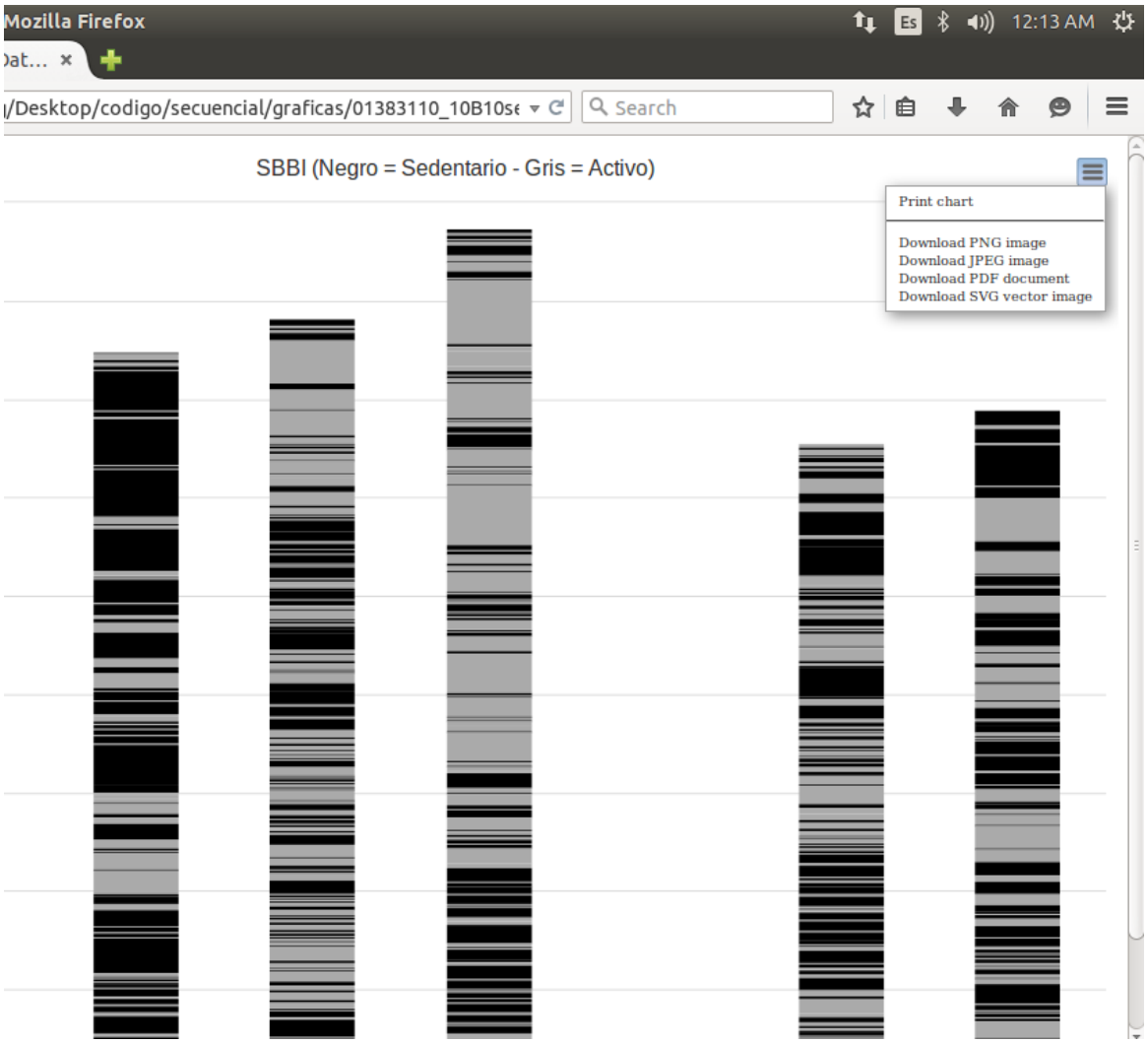


Ilustración 35. Exportación de la gráfica.

Una vez abierto, se podrá elegir el formato de salida deseado. El fichero será descargado por el navegador, pudiendo mostrar la ventana para abrirlo o guardarlo en un directorio.

LIMPIEZA

El fichero makefile tiene una regla que limpia los ficheros generados tras la ejecución para hacer más ligero el proyecto y poder transportarlo en un pen drive sin preocuparse por el espacio. Una vez terminadas las ejecuciones deseadas y guardar las gráficas que nos interesan, se puede llamar a la regla “make clean” que eliminará los objetos generados por la compilación inicial, las gráficas generales y los ficheros HTML de la carpeta “GRAFICAS”.

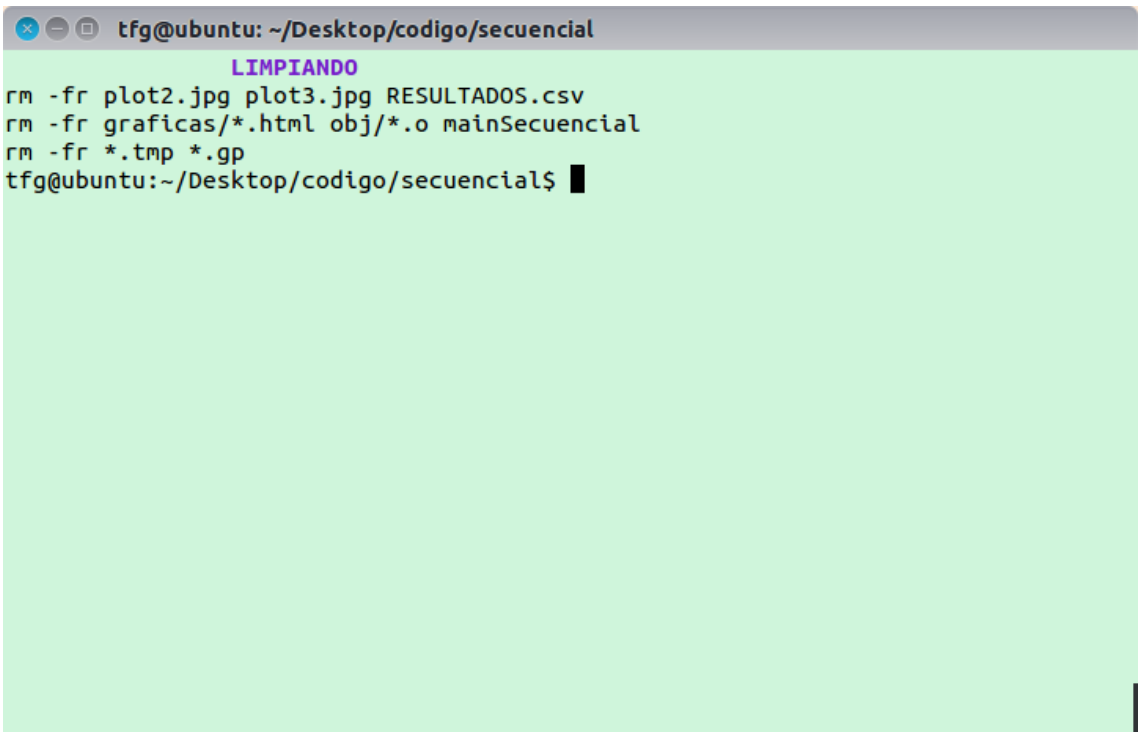


Ilustración 36. Llamada a la regla "clean".

ANEXO B. Tablas de los sujetos originales procesados con distribución = 1.

TIEMPO			
SERIE	2 CORES	3 CORES	4 CORES
0,3773	0,4608	0,3925	0,3883
0,6263	0,6128	0,6311	0,6047
1,4715	1,3782	1,3624	1,372
2,6824	2,5205	2,507	2,4958
5,6977	4,8788	5,0116	4,8436
10,979	10,8206	11,0695	9,9743
23,0673	20,0438	20,4167	20,2195
33,8396	31,0357	30,8592	30,5173

SPEED-UP - TIEMPO		
2 CORES	3 CORES	4 CORES
0,8188	0,9613	0,9717
1,0220	0,9924	1,0357
1,0677	1,0801	1,0725
1,0642	1,0700	1,0748
1,1678	1,1369	1,1763
1,0146	0,9918	1,1007
1,1508	1,1298	1,1408
1,0903	1,0966	1,1089

SPEED-UP = Ts / Tp	
1000 files	SPEED-UP
2 CORES	1,0903
3 CORES	1,0966
4 CORES	1,1089

*OVERHEAD = p * Tp - Ts*	
1000 files	OVERHEAD
2 CORES	28,2318
3 CORES	58,7380
4 CORES	88,2296

EFICIENCIA =SpeedUp / p	
1000 files	EFICIENCIA
2 CORES	0,5452
3 CORES	0,3655
4 CORES	0,2772

OVERHEAD - TIEMPO		
2 CORES	3 CORES	4 CORES
0,5443	0,8002	1,1759
0,5993	1,2670	1,7925
1,2849	2,6157	4,0165
2,3586	4,8386	7,3008
4,0599	9,3371	13,6767
10,6622	22,2295	28,9182
17,0203	38,1828	57,8107
28,2318	58,7380	88,2296

*COSTE = p * Tp*	
1000 FILES	COSTE
2 CORES	62,0714
3 CORES	92,5776
4 CORES	122,0692

ANEXO C. Tablas de los sujetos originales procesados con distribución = 10.

TIEMPO			
SERIE	2 CORES	3 CORES	4 CORES
0,3773	0,4027	0,4021	0,4036
0,6263	0,6318	0,6484	0,6463
1,4715	1,4059	1,3569	1,3712
2,6824	2,7022	2,5254	2,5438
5,6977	5,1629	4,8794	4,9439
10,979	10,8206	11,0695	9,6393
23,0673	20,6711	20,6768	20,6948
33,8396	30,8357	30,7532	30,4723

SPEED-UP - TIEMPO		
2 CORES	3 CORES	4 CORES
0,9369	0,9383	0,9348
0,9913	0,9659	0,9691
1,0467	1,0845	1,0731
0,9927	1,0622	1,0545
1,1036	1,1677	1,1525
1,0146	0,9918	1,1390
1,1159	1,1156	1,1146
1,0974	1,1004	1,1105

SPEED-UP = Ts / Tp	
1000 FILE	SPEED-UP
2 CORES	1,0974
3 CORES	1,1004
4 CORES	1,1105

*OVERHEAD = p * Tp - Ts*	
1000 FILES	OVERHEAD
2 CORES	27,8318
3 CORES	58,4200
4 CORES	88,0496

EFICIENCIA =SpeedUp / p	
1000 FILE	EFICIENCIA
2 CORES	0,5487
3 CORES	0,3668
4 CORES	0,2776

OVERHEAD - TIEMPO		
2 CORES	3 CORES	4 CORES
0,4281	0,8290	1,2371
0,6373	1,3189	1,9589
1,3403	2,5992	4,0133
2,7220	4,8938	7,4928
4,6281	8,9405	14,0779
10,6622	22,2295	27,5782
18,2749	38,9631	59,7119
27,8318	58,4200	88,0496

*COSTE = p * Tp*	
1000 FILES	COSTE
2 CORES	61,6714
3 CORES	92,2596
4 CORES	121,8892

ANEXO D. Tablas de los sujetos válidos procesados con distribución = 10.

TIEMPO			
SERIE	2 CORES	3 CORES	4 CORES
0,4597	0,4345	0,4324	0,4439
0,7012	0,5982	0,625	0,658
1,0541	0,88	0,7968	0,7967
3,056	2,5796	2,5031	1,9806
5,8271	5,002	4,7194	4,9052
14,6667	12,8072	12,1504	11,4854
26,7342	22,5579	22,7092	22,3502
38,9789	32,229	31,2104	31,4333

SPEED-UP - TIEMPO		
2 CORES	3 CORES	4 CORES
1,0580	1,0631	1,0356
1,1722	1,1219	1,0657
1,1978	1,3229	1,3231
1,1847	1,2209	1,5430
1,1650	1,2347	1,1879
1,1452	1,2071	1,2770
1,1851	1,1772	1,1962
1,2094	1,2489	1,2401

SPEED-UP = Ts / Tp	
1000 FILE	SPEED-UP
2 CORES	1,2094
3 CORES	1,2489
4 CORES	1,2401

*OVERHEAD = p * Tp - Ts*	
1000 FILES	OVERHEAD
2 CORES	25,4791
3 CORES	54,6523
4 CORES	86,7543

EFICIENCIA =SpeedUp / p	
1000 FILE	EFICIENCIA
2 CORES	0,6047
3 CORES	0,4163
4 CORES	0,3100

OVERHEAD - TIEMPO		
2 CORES	3 CORES	4 CORES
0,4093	0,8375	1,3159
0,4952	1,1738	1,9308
0,7059	1,3363	2,1327
2,1032	4,4533	4,8664
4,1769	8,3311	13,7937
10,9477	21,7845	31,2749
18,3816	41,3934	62,6666
25,4791	54,6523	86,7543

*COSTE = p * Tp*	
1000 FILE	COSTE
2 CORES	64,4580
3 CORES	93,6312
4 CORES	125,7332